

Tự Học Microsoft ASP.NET

Bản quyền của:
www.vovisoft.com

Rào Trước

Hành trang vào Khóa Học ASP.NET

Ta nên chuẩn bị sẵn một số kiến thức căn bản về lập trình hay phát triển mạng khi bước vào khóa học này thì tốt hơn, vì tuy bạn không cần phải biết về ASP cổ điển (classic ASP) nhưng, như chúng tôi đã trình bày trong phần FAQ ở trang Chào Mừng đầu khóa học, bạn cần:

- tham khảo các bài viết trong khóa [Học Microsoft .NET](#) của [thầy Lê Đức Hồng](#) để làm quen với .NET framework, Visual Basic.NET, Visual Studio.NET. VB.NET sẽ được dùng làm ngôn ngữ mặc định (default) trong các thí dụ, các bài tập hay các dự án của khóa.
- biết tổng quát về HTML (HyperText Markup Language) khi ta cần trình bày các trang web trên browser. Browser được dùng trong khóa này là IE6 (Internet Explorer Version 6).
- quen thuộc với các hệ điều hành mới hiện nay (Operating System) như Windows 2000 (Professional hay Server) hay Windows XP (Home hay Professional), cũng như quen thuộc cách quản lý các ứng dụng liên hệ như Web Server (Personal Web Server hay Internet Information Server - IIS) và các cơ sở dữ liệu (database) MS SQL Server 2000 - xin tham khảo các bài viết về [MCSE](#) của [thầy Vũ Hữu Tín](#), [thầy Tăng Vinh Tài](#) và [lớp MCSE](#).
- vài kiến thức căn bản về XML liên quan đến việc chuyển thông tin từ chỗ này qua chỗ khác. Về XML, bạn nên tham khảo các bài viết [XML, Kỹ Thuật Nòng Cốt trong Tương Lai](#) của thầy [Lê Đức Hồng](#) và các [bài tự học XML](#) của [cô Bạch Trí](#) cũng trên mạng Vovisoft này).

Thật ra, ta đâu làm khó nhau chi nhưng phải rào trước đón sau như vậy là vì con đường ta đi tìm hiểu về ASP.NET hơi lắt léo gập ghềnh. Một khi ta phát triển mạng với ASP.NET, ta phải vận dụng tất cả các ứng dụng liên hệ và kết hợp mọi thứ vào nhau. Đó cũng là lý do tại sao khóa học này chỉ nhắm vào các lập trình viên đã có kinh nghiệm phát triển mạng. Tuy vậy, chúng tôi sẽ cố gắng trình bày một cách đơn sơ, ngắn và gọn khi đề cập đến các ứng dụng kể trên trong các bài học có liên quan đến để bạn (nhất là bạn nào thích thú trong việc phát triển mạng và mới làm quen với ASP.NET) dễ dàng theo dõi và tìm hiểu về ASP.NET.

Nhu Liệu (Software)

Nhu liệu (phần mềm hay software) tối thiểu phải có để học khóa ASP.NET thành công:

- Về hệ điều hành (Operating System): ta nên dùng Windows 2000 (Professional hay Server) hay Windows XP (Home hay Professional). Lý do chính là các versions của Windows này hỗ trợ Unicode và có thể cài tự do Internet Information Server (IIS Version 5) hỗ trợ ASPX để ta dùng cho ASP.NET vì IIS được cung cấp miễn phí trong các hệ điều hành kể trên.
- Ngoài ra, bạn cần phải có .NET framework SDK (Software Development Kit) tải xuống từ mạng Microsoft. Chúng tôi sẽ trình bày việc tải xuống ở đâu và công cuộc bố trí như thế nào ở bài học số 1. Nếu như bạn đã có và cài đặt Microsoft Visual Studio.Net (VS.NET) theo sự hướng dẫn các bài học của khóa Học Microsoft .NET thì càng tốt nữa. Vì rắc rối trong các bài học về ASP.NET, chúng tôi cũng trình bày một số thí dụ dùng VS.NET cho ASP.NET mặc dù thật ra chỉ cần có Notepad thôi đã cũng ... dư xài suốt năm.
- Cuối cùng, bạn cần phải có và cài đặt thành công một cơ sở dữ liệu (Database) vững mạnh tương ứng với OLE DB-compliant database system như SQL Server 2000 để dùng lưu trữ những thông tin cần thiết. Tuy nhiên, ta cũng có thể dùng MS Access 2000 để tạm thời thay thế SQL Server 2000 cho các thí dụ trong khóa phần nhiều trình bày các nối và các kiểu thu thập dữ liệu hay thông tin xuyên qua SQL Server 2000.

Tại sao ta lại quan tâm và phát triển mạng với ASP.NET

Ta phải công nhận một điều là .NET Framework và các ứng dụng của nó đã và đang tạo một cuộc cách mạng kỹ thuật trong công nghệ Tin Học (Information Technology), thay đổi tận gốc rễ các kiểu mẫu lập trình hay phát triển và triển khai mạng trên thế giới và do đó tạo một vận hội mới đáp ứng mọi yêu cầu khẩn thiết cho các ngành nghề kỹ thuật và thương mại hiện nay cũng như vạch một hướng đi vững chắc và dài lâu cho tương lai Tin Học. ASP.NET chính là một trong những ứng dụng quan trọng nhất để phát triển và triển khai mạng một cách dễ dàng chưa từng ... thấy từ xưa đến nay. Thật vậy, không phải là

chúng tôi khoái ... nó đâu, hãy lắng nghe thử chính Microsoft đã nói về ASP.NET như thế nào:

'ASP.NET is a revolutionary programming framework that enables the rapid development of powerful web applications and services. Part of the Microsoft .NET Platform, it provides the easiest and most scalable way to develop, deploy and run distributed web applications that can target any browser or any application.'

Có ý chang như vậy hay không? hay là chỉ quảng cáo theo kiểu ... 'cao đon hườn tán' mà thôi? Chúng ta hãy ... 'ngồi xuống đây, hãy ngồi xuống đây' và ngồi gần lại với nhau để nhìn cho rõ những đặc tính không ... tầm thường của ASP.NET như sau:

Developer Productivity

Easy Programming Model:

ASP.NET giúp ta phát triển và triển khai các ứng dụng về mạng trong một thời gian kỷ lục vì nó cung cấp cho ta một kiểu mẫu lập trình dễ dàng và gọn gàng nhất. Ngoài ra còn bảnh hơn nữa, các trang ASP.NET làm việc với mọi browsers hiện nay như Internet Explorer (IE), Netscape, Opera, AOL, ... mà không cần phải đối tời đối lui các nguồn mã rất vất vả như trước.

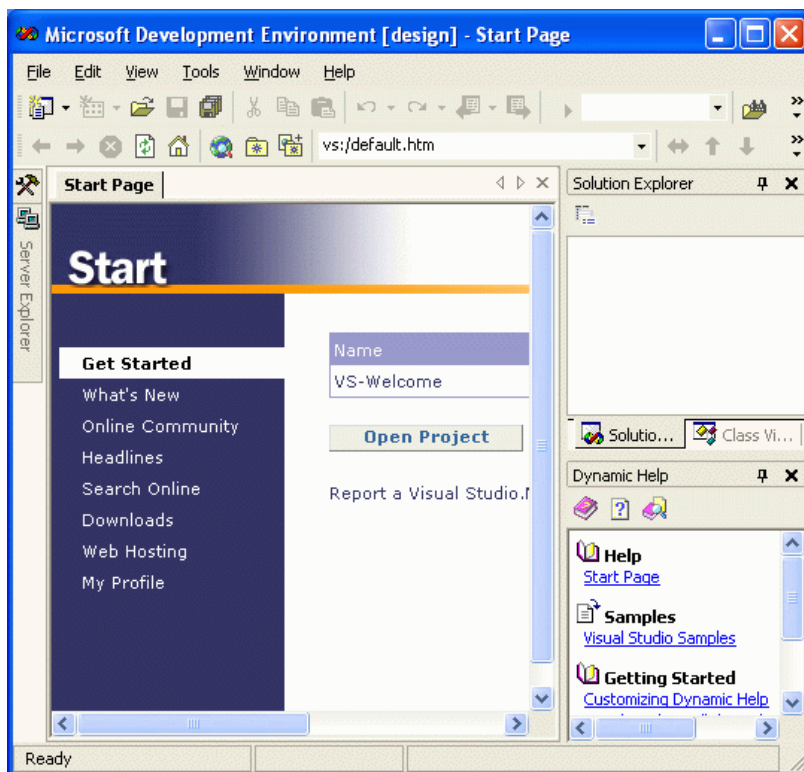
Flexible Language Options:

Không như ASP kiểu cổ điển chỉ giới hạn với VBScripts and JScripts, ASP.NET yểm trợ trên 25 .NET ngôn ngữ lập trình (dĩ nhiên ngoài các ngôn ngữ mới thiết lập đã cài sẵn yểm trợ .NET framework như là VB.NET, C# và JScript.NET còn có MC++.NET, Smalltalk.NET, COBOL.NET, Eiffel.NET, Perl.NET, Component Pascal.NET, Mercury.NET, Oberon.NET, Python.NET, vân vân và vân vân ... không kể xiết).

Great Tool Support:

Mặc dù ta có thể chỉ cần dùng tới Notepad để triển khai các trang ASP.NET nhưng Visual Studio.NET giúp năng suất triển khai mạng thêm phần hiệu quả ví ta có thể quan sát các kế hoạch của ta dễ dàng hơn khi phát họa (design) các thành phần của ASP.NET bằng hình ảnh với ASP.NET Web Forms hay Services theo phương pháp 'drag-drop-doubleclick' quen thuộc của nền Windows. Thêm nữa, lại còn yểm trợ ta trong việc phát hiện và loại bỏ những lỗi sai một cách rất thuận lợi trong khi phát triển các ứng dụng về mạng (support for debugging and deploying ASP.NET Web applications).

Đây là trang đầu tiên khi dùng Visual Studio.NET:



Rich Class Framework:

Nhờ nền tảng vững vàng và tài nguyên phong phú của .NET Framework với hơn 5000 classes bao gồm đủ thứ mọi chuyện trên trời dưới đất như XML, data access, file upload, regular expressions, transactions, message queuing, SMTP mail, vân vân và vân vân ... nên việc thiết kế các đặc tính trong một ứng dụng trở nên nhẹ nhàng và thoải mái hơn xưa rất nhiều.

Improved Performance and Scalability

Compile execution:

ASP.NET không những chạy nhanh hơn ASP cổ điển gấp 5 lần mà còn có thể duy trì kiểu mẫu cập nhật gọi là kiểu mẫu '**just hit save**', nghĩa là ASP.NET tự động dò tìm mọi sự thay đổi và compile files khi cần thiết cũng như lưu trữ kết quả compile đó để cung ứng dịch vụ cho những yêu cầu tiếp theo sau, nhờ vậy ứng dụng của bạn luôn luôn cập nhật hóa và chạy nhanh hơn cả ... ngựa Nhâm Ngọ 2002.

Rich output caching:

ASP.NET có khả năng lưu trữ một kết quả chung trong phân bộ memory của trang để gởi giải đáp cho cùng một yêu cầu từ nhiều khách hàng khác nhau và nhờ đó không những tiết kiệm được sự lập đi lập lại công tác thi hành của một trang web mà còn gia tăng hiệu xuất một cách ngoạn mục do giới hạn tối đa việc chất vấn các cơ sở dữ liệu (eliminating the need to query the database on every request) rất tốn nhiều thời gian.

.NET Outperforms J2EE:

Trong việc đối đầu với nhau về hiệu xuất (performance) và scalability với cùng một ứng dụng phát triển giữa Sun's Java Pet Store J2EE và ASP.NET thì ASP.NET không những nhanh vượt trội hơn J2EE đến 28 lần (khoảng 2700%), nguồn mã lại ít hơn nhiều (khoảng 1/4 nguồn mã của J2EE) mà còn dùng bộ xử lý (processor) chỉ khoảng 1/6 lần so với việc sử dụng processor của J2EE. Nếu muốn tìm hiểu thêm, bạn có thể truy tìm dễ dàng những trang so sánh như vậy khi nối vào Internet để 'search' về hiệu xuất của cả hai sản phẩm.

Enhanced Reliability

Memory Leak, DeadLock và Crash protection:

ASP.NET cũng có khả năng tự động dò tìm và phục hồi (detects and recovers) những trở ngại nghiêm trọng như deadlocks hay bộ nhớ (memory) bị rì để bảo đảm ứng dụng của bạn luôn luôn sẵn sàng khi dùng mà không làm cản trở việc cung ứng dịch vụ cần thiết thường lệ.

Easy Deployment

Simple application deployment:

ASP.NET đơn giản hóa việc triển khai ứng dụng mạng, do đó biến việc triển khai toàn bộ ứng dụng trở nên dễ dàng và thuận lợi hơn hẳn trước kia vì bây giờ ta chỉ cần sao (với XCOPY) và lưu trữ ở Server chứ không cần phải chạy chương trình 'regsrv32' để đăng ký bất cứ thành phần nào cả, và thêm nữa, khi cần lưu trữ những yếu tố phụ cần thiết cho việc thiết lập hay bố trí các ứng dụng, ta chỉ cần lưu giữ nó vào trong một hồ sơ dưới dạng XML là đủ.

Dynamic update of running application:

ASP.NET cho phép ta tự động cập nhật hóa (update) các thành phần đã compiled (compiled components) mà không cần phải khởi động lại (re-start) các Web Server.

Easy Migration Path:

Ta không cần phải du nhập những ứng dụng được phát triển và triển khai bằng ASP cổ điển hiện có của bạn vào ASP.NET vì ASP.NET có thể chạy song song với ... cừu chiến binh ASP ở cùng một Internet Information Server (IIS) trong nền Windows 2000 hay nền Windows XP. Các ứng dụng cũ vẫn tiếp tục chạy hết sức thoải mái với ASP.DLL trong khi ASP.NET engine sẽ xử lý các ứng dụng mới. Ngoài ra, ASP.NET còn cho phép bạn dùng lại những thành phần thương mại hiện nay kiểu COM cổ điển trong các ứng dụng của nó.

XML Web Services:

Dịch vụ tân kỳ về mạng với XML cũng cho phép bạn truyền đạt (communicate) và chia sẻ (share) các dữ kiện (data) xuyên qua mạng Internet dễ dàng tới các SOAP client mà không hề phân biệt đối xử các hệ điều hành hay các ngôn ngữ lập trình khác nhau (regardless of OS or programming language). Nhờ đó, ta không cần phải học thêm hay đào sâu các kiến thức về Networking, XML hay SOAP, ...

Mobile Web Service Support:

Thêm nữa, ASP.NET Mobile Controls còn giúp ta phát triển và triển khai mạng nhắm vào thị trường những cell phone hay PDA với gần hơn 80 Mobile Web Services được cung cấp trong .NET framework. Bạn chỉ cần lập trình cho ứng dụng của bạn như thường lệ rồi phó mặc cho Mobile Controls đó tự động phát sinh ra những nguồn mã như WAP/WML, HTML hay iMode thích hợp với từng loại thiết bị (device) riêng biệt.

Thôi, đến đây ta tạm ngừng ... 'đánh bóng' những gì đã sáng chói đó mà hít một hơi thật dài, bước một bước khởi đầu cho cuộc hành trình tìm hiểu và thực hành về ASP.NET. Vậy xin mời bạn hãy sang sông, à không, sang ... trang với bài số 1 **'Làm Quen với ASP.NET'**

Bài 01

Làm Quen với ASP.NET

*Hãy ngồi xuống đây, hãy ngồi xuống đây, xa cơn buồn phiền.
Đầu biết chia phôi, nhưng trong cuộc đời, vẫn có đôi ta.
Hãy ngồi xuống đây - Lê Uyên Phương*

Trong trang này, ta sẽ tham khảo một số vấn đề như sau:

- ASP.NET là gì?
- Phương pháp làm việc trong mạng
- Sơ lược về .NET Framework
- Bố trí và cài đặt ASP.NET
- Tạo trang ASP.NET đầu tiên

Chắc hẳn ta có dịp được nghe Nguyễn Ngọc Ngạn kể với Kỳ Duyên rằng ông đã bỏ công gần 2, 3 năm trời đăng đăng để ... nghiên cứu rất là cực khổ mới khám phá ra một chân lý như sau: 'Thà là lấy vợ vừa trẻ vừa đẹp còn hơn là lấy vợ ... vừa già vừa xấu'. Ta thì không cần phải tốn nhiều năm tháng đến thế để biết được ASP.NET vừa trẻ vừa đẹp hơn là ASP hoặc những phương pháp phát triển mạng cổ điển khác nhưng dù có biết rõ ASP.NET vừa trẻ vừa đẹp (Microsoft chính thức khai trương sản phẩm .NET Framework và MS Visual studio.NET vào ngày 13 tháng Hai năm 2002, tuy nhiên rất nhiều phiên bản Beta2 đã được phổ biến khắp thế giới giữa năm 2001) nhưng mấy ai biết được ASP.NET là gì và dung nhan ... mùa hạ ra làm sao? Vậy ta 'hãy ngồi xuống đây' để bắt đầu làm quen với ASP.NET.

ASP.NET - EM LÀ AI?

Trước hết, họ tên của ASP.NET là Active Server Pages .NET (.NET ở đây là .NET framework). Nói đơn giản, ngắn và gọn thì ASP.NET là một công nghệ có tính cách mạng dùng để phát triển các ứng dụng về mạng hiện nay cũng như trong tương lai (ASP.NET is a revolutionary technology for developing web applications). Bạn lưu ý ở chỗ ASP.NET là một phương pháp tổ chức hay khung tổ chức (framework) để thiết lập các ứng dụng hết sức hùng mạnh cho mạng dựa trên CLR (Common Language Runtime) chứ không phải là một ngôn ngữ lập trình. Ngôn ngữ lập trình được dùng để diễn đạt ASP.NET trong khóa này là VB.NET (Visual Basic .NET) và VB.NET chỉ là một trong 25 ngôn ngữ .NET hiện nay được dùng để phát triển các trang ASP.NET mà thôi.

Tuy mang họ tên gần giống như ASP cổ điển nhưng ASP.NET không phải là ASP. Ta sơ lược ở đây vài khác biệt giữa ASP.NET và ASP để bạn có khái niệm tổng quát và sẽ trình bày thêm chi tiết khi đào sâu vào từng điểm đặc trưng (features) của ASP.NET ở từng bài học một.

KHÁC BIỆT GIỮA ASP.NET VÀ ASP

ASP.NET được phác thảo (re-design) lại từ số không, nó được thay đổi tận gốc rễ và phát triển (develop) phù hợp với yêu cầu hiện nay cũng như vạch một hướng đi vững chắc cho tương lai Tin Học. Lý do chính là Microsoft đã quá chán nản trong việc thêm thắt và kết hợp các công dụng mới vào các kiểu mẫu lập trình hay thiết kế mạng theo kiểu cổ điển nên Microsoft nghĩ rằng tốt nhất là làm lại một kiểu mẫu hoàn toàn mới thay vì vá vúi chỗ này chỗ nọ vào ASP. Đó là chưa kể đến nhiều phát minh mới ra đời sau này dựa trên các khái niệm mới mẽ theo xu hướng phát triển hiện nay của công nghệ Tin Học (Information Technology) cần được đưa vào kiểu mẫu phát triển mới đó. Nhờ vậy, ta mới có thể nói ... khơi khơi ASP.NET không phải là ASP. Thật vậy, ASP.NET cung cấp một phương pháp hoàn toàn khác biệt với phương pháp của ASP.

Lưu ý, mặc dù ASP.NET và ASP khác biệt nhau nhưng chúng có thể hoạt động vui vẻ hài hoà với nhau trong Web Server của bạn (operate side-by-side). Do đó, khi bạn cài ASP.NET engine, bạn không cần lập trình lại các ứng dụng hiện có dưới dạng ASP của bạn tuy rằng, nếu muốn, bạn có thể làm điều đó rất dễ dàng.

SỰ THAY ĐỔI CƠ BẢN

ASP đã và đang thi hành sứ mạng được giao cho nó để phát triển mạng một cách tốt đẹp như vậy thì tại sao ta cần phải đổi mới hoàn toàn? Lý do đơn giản là ASP không còn đáp ứng đủ nhu cầu hiện nay trong lãnh vực phát triển mạng của công nghệ Tin Học. ASP

được thiết kế riêng biệt và nằm ở tầng phía trên hệ điều hành Windows và Internet Information Server, do đó các công dụng của nó hết sức rời rạc và giới hạn.

Trong khi đó, ASP.NET là một cơ cấu trong các cơ cấu của hệ điều hành Windows dưới dạng nền hay khung .NET (.NET framework), như vậy ASP.NET không những có thể dùng các object của các ứng dụng cũ mà còn có thể xử dụng tất cả mọi tài nguyên mà Windows có, dễ dàng như ... ăn cơm tắm bị sườn chả vậy.

Ta có thể tóm tắt đại khái sự thay đổi như sau:

- Tập tin của ASP.NET (ASP.NET file) có extension là .ASPX, còn tập tin của ASP là .ASP.
- Tập tin của ASP.NET (ASP.NET file) được phân tích ngữ pháp (parsed) bởi XSPISAPI.DLL, còn tập tin của ASP được phân tích bởi ASP.DLL.
- ASP.NET là kiểu mẫu lập trình phát động bằng sự kiện (event driven), còn các trang ASP được thi hành theo thứ tự tuần tự từ trên xuống dưới.
- ASP.NET xử dụng trình biên dịch (compiled code) nên rất nhanh, còn ASP dùng trình thông dịch (interpreted code) do đó hiệu suất và tốc độ phát triển cũng thua sút hẳn.
- ASP.NET yểm trợ gần 25 ngôn ngữ lập trình mới với .NET và chạy trong môi trường biên dịch (compiled environment), còn ASP chỉ chấp nhận VBScript và JavaScript nên ASP chỉ là một scripted language trong môi trường thông dịch (in the interpreter environment). Không những vậy, ASP.NET còn kết hợp nhuần nhuyễn với XML (Extensible Markup Language) để chuyển vận các thông tin (information) qua mạng.
- ASP.NET yểm trợ tất cả các browser và quan trọng hơn nữa là yểm trợ các thiết bị lưu động (mobile devices). Chính các thiết bị lưu động, mà mỗi ngày càng phổ biến, đã khiến việc dùng ASP trong việc phát triển mạng nhằm vươn tới thị trường mới đó trở nên vô cùng khó khăn.

PHƯƠNG PHÁP LÀM VIỆC TRONG MẠNG

Internet đã và đang đem lại nhiều điều kỳ diệu cho đời sống của ta. Thật vậy, nó có khả năng 'nối vòng tay lớn' mọi người trên thế giới tưởng chừng như cách biệt xa xôi ngàn dặm bỗng dung lại gần trong gang tấc, kỹ thuật này đã mang lại biết bao nhiêu điều mới mẻ đến cho ta tỷ như e-mail, instant messaging hay World Wide Web (hay gọi tắt là WWW hay Web hay mạng) làm việc thông tin liên lạc trở nên dễ dàng, do đó con người cùng đời sống cũng thay đổi nhanh chóng như ...'cuốn theo chiều gió'.

Từ khởi đầu, việc phát triển 1 mạng hết sức là đơn giản, chỉ cần một hay vài trang trong đó ta muốn chia sẻ bất cứ thông tin gì ta thích là chắc chắn cũng có người ghé qua thăm viếng. Tuy vậy, các trang trong thời kỳ khởi nguyên của mạng rất thụ động, nó không cho phép khách vãng lai trao đổi thông tin một cách hô tương (interact) với ta, nghĩa là thăm thì có thăm nhưng không hỏi hay chia sẻ được gì với nhau.

Dần dà, mạng phát triển thêm nhiều công dụng khác nhau gắn thêm vào nào là hình ảnh, nào là tables, forms và cuối cùng có thể trao đổi thông tin hay tâm tình với khách vãng lai qua các ứng dụng như guestbook, thăm dò ý kiến (user, customer hoặc là client poll) hay các diễn đàn với mọi tiết mục trên trời dưới đất. Sau đó, các chuyên gia phát triển mạng lại thêm thắt và trang điểm cho mạng của mình càng lúc càng đặc sắc hơn, càng muôn màu muôn vẻ.

Tất cả những cố gắng đó đã đem tác động hỗ tương đến giữa Web Master (hay nhóm quản lý mạng) và khách vãng lai như ta được chứng kiến hiện nay, tuy vậy vẫn còn thiếu hẳn 1 phần quan trọng nhất là phần nội dung cơ động tùy biến (dynamic content). Do đó vai trò của phương pháp dịch vụ (server processing) được phát triển để có thể trình bày nội dung được lưu trữ trong các cơ sở dữ liệu (database) tùy theo yêu cầu riêng biệt cho từng cá nhân.

Kiểu MẪU REQUEST/RESPONSE

Kiểu mẫu này chính là toàn bộ phương pháp làm việc theo kiểu Client /Server hiện dùng với ASP.

Client/Server - Một trường hợp đơn giản nhất là sự trao đổi thông tin giữa 2 máy vi tính để hoàn thành 1 công việc được định trước, trong đó máy Server cung cấp dịch vụ theo yêu cầu của 1 máy khách hành (Client PC).

Thường thường, Server là máy vi tính lưu trữ thông tin về mạng trong đó có hình ảnh, video, những trang HTML hay ASP. Client là máy vi tính được dùng để lướt mạng. Một cách tổng quát phương pháp này gồm có 4 bước cơ bản sau:

1. Client (thông qua Internet Browser) xác định vị trí của Web Server qua 1 nối URL (**Universal Resource Locator**) tỷ như www.vovisoft.com
2. Client sẽ yêu cầu được tham khảo 1 trang trong mạng đó và thường là trang chủ (home page) tỷ như index.htm hay default.htm

3. Server đáp ứng bằng cách hoàn trả hồ sơ mà Client đã yêu cầu.
4. Client nhận được hồ sơ gửi về và hiển thị (display) trong browser của mình.

Lưu ý, một khi Client đã nhận được hồ sơ rồi, quá trình trao đổi qua lại đó kết thúc ngay tức khắc. Sau đó, Server và Client trở thành ... 'người xa lạ', coi như là chưa từng bao giờ gặp nhau (stateless model), ta gọi là kiểu ... 'làm ngo'.

KIỂU MẪU EVENT-DRIVEN

Kiểu mẫu event-driven này dùng với ASP.NET cũng tương tự như là kiểu mẫu event-driven mà ta vẫn thường dùng trong khi lập trình các ứng dụng với Visual Basic 6.

Trong kiểu mẫu này, Server sẽ không 'ngồi ... chơi xơi nước' chờ Client yêu cầu tham khảo 1 trang nào đó trong mạng mà Server đã bố trí và kế hoạch sẵn trước tất cả mọi tình huống để có thể hành động kịp thời mỗi khi Client quyết định làm 1 điều gì đó. Ta gọi đó là 'response to your action', còn trong kiểu mẫu trước là 'response to your request', như vậy ASP.NET có thể phát hiện ra các hành động của Client để phản ứng cho thích hợp.

Đọc tới đây chắc bạn sẽ hỏi lại ngay rằng: 'Ừa, nhưng mà làm sao một Server nào đó, có thể ở tận đâu đâu bên kia địa cầu, lại biết được là ta đang gõ vài mẩu tự trong một hộp chữ hay là đang nhấp mũi chuột (click) vào button trong phần Guestbook hay Forum của Vovisoft để gửi đi một thông điệp làm quen với Vovisoft?'.

À, sở dĩ Server có thể làm được 'chuyện ... khó tin nhưng có thiệt đó' là dựa vào tiến trình xử lý linh động ở Client (gọi là clever client-side processing) để thực hiện kiểu mẫu event-driven này của mình. Tiến trình xử lý ở Client xảy ra khi ta bố trí nguồn mã thích hợp mà Client có thể hiểu được trong các trang ta gửi về cho Client. Lưu ý là mặc dù các trang mạng (web page) ta đều chứa ở Server nhưng nguồn mã lại có thể được thực hiện và xử lý, hoặc ở Server hoặc ở Client (Server-Side processing và Client-Side processing) tùy theo cách ta bố trí. Thật vậy, ASP.NET không thể nào biết được chuyện gì sẽ xảy ra ở máy vi tính của bạn (Client PC) nhưng nhờ vào tiến trình xử lý linh động ở Client mà Server có thể tiến hành kiểu mẫu phát triển mạng mới theo phương pháp event-driven.

Nhớ là ta có thể chạy nguồn mã ở 2 chỗ khác nhau: hoặc là chạy ở Server (gọi là Server-side) hoặc là chạy ở Client (Client-side) và các nguồn mã ở 2 chỗ này hoàn toàn khác biệt, không có tác động hỗ tương với nhau (no interact with each other). Điều đó có nghĩa là máy Client sẽ chịu trách nhiệm thi hành các nguồn mã được lập trình dành cho mình cũng như máy Server chỉ chạy các nguồn mã dành cho Server. Thông tin hay nội dung cần thiết ở Server sẽ được chuyển sang dạng HTML đơn giản (plain HTML) trước khi gửi đến cho Client, thường thì nguồn mã dành cho Client cũng được chuyển đi dưới dạng 'plain text command' để thực hiện các hiệu ứng năng động (dynamic effect) ở máy Client, tỷ như thay đổi hình ảnh (image rollover) hay hiển thị một thông điệp (message box).

ASP.NET sẽ dùng các ngôn ngữ mới có trình biên dịch (compiled languages) như C# hay VB.NET để soạn các nguồn mã trong các trang Web ở Server.

SƠ LƯỢC VỀ .NET FRAMEWORK

Mọi chức năng ASP.NET có được hoàn toàn dựa vào .NET framework, do đó có chữ .NET trong ASP.NET. Ta cần phải hiểu thấu đáo kiến trúc hạ tầng của .NET framework để dùng ASP.NET một cách hiệu quả, trong đó quan trọng nhất là **CLR** và **.NET Framework Class**.

CLR (COMMON LANGUAGE RUNTIME)

CLR là môi trường được dùng để quản lý sự thi hành các nguồn mã (manage the execution of code) mà ta đã soạn ra và biên dịch (write and compile code) trong các ứng dụng. Tuy nhiên khi biên dịch nguồn mã, ta lại biên dịch chúng ra thành một ngôn ngữ trung gian gọi là **Microsoft Intermediate Language (MSIL)**. Chính MSIL trung gian này là ngôn ngữ chung cho tất cả các ngôn ngữ .NET hiện có, do đó chắc bạn cũng đoán ra là ASP.NET cũng được biên dịch (compile) ra MSIL như mọi ai khác. Trong khi biên dịch như vậy, các ứng dụng cũng sản xuất ra những thông tin cần thiết để tự ... quảng cáo chính mình, ta gọi những thông tin này là **metadata**. Đến khi ta chạy một ứng dụng, CLR sẽ tiếp quản (take-over) và lại biên dịch (compile) nguồn mã một lần nữa ra thành ngôn ngữ gốc (native language) của máy vi tính trước khi thi hành những công tác đã được bố trí trong nguồn mã đó. Ta có thể cảm thấy những việc bận rộn sau hậu trường đó khi phải chờ đợi 1 khoảng thời gian cần thiết để CLR chấm dứt nhiệm vụ của nó khi lần đầu phải biên dịch (compile) và hiển thị 1 trang Web, nhưng rồi mọi chuyện sẽ xuôi chèo mát mái, cuối cùng là ta có một trình biên dịch (compiled code) để xử dụng rất hiệu quả.

.NET FRAMEWORK CLASSES

Điều quan trọng nhất mà ta cần phải nhớ là mọi thứ trong .NET đều là **object**, tỷ như các trang ASP.NET, các hộp thông điệp (message box) hay là nút bấm (button), tất cả đều là object cả. Các object đó được tổ chức lại thành từng nhóm riêng biệt như trong một thư viện để ta dễ dàng sử dụng. Ta gọi các nhóm như vậy là **namespaces**, và ta sẽ dùng những namespace này để gọi hay nhập (import) các class cần thiết cho ứng dụng của mình.

Ở đây, ta chỉ sơ lược một chút về .NET framework mà thôi, bạn có thể tham khảo đầy đủ chi tiết về .NET framework ở các bài viết của **thầy Lê Đức Hồng** trong khóa Học **[.NET Framework và VB.NET](#)**

CÀI ĐẶT ASP.NET

Để chạy trang ASP.NET, trước hết ta cần phải cài đặt thành công:

- **Internet Information Server (IIS)** và bố trí Virtual Directory dùng trong khóa Tự Học ASP.NET của Vovisoft.
- **MS Visual Studio.NET** - trong trường hợp này thì MS Visual Studio.NET đã cài sẵn .NET Framework SDK cho ta dùng với ASP.NET hoặc là Microsoft ASP.NET Web Matrix (chi tiết được trình bày ở bài **Giới Thiệu Về Web Matrix**).
- **.NET Framework Software Development Kit (SDK)** - nếu ta không có MS Visual Studio.NET, ta có thể tải .NET Framework Software Development Kit (SDK) xuống tự do từ mạng www.microsoft.com/.NET, với SDK, ta chỉ có thể dùng Notepad hoặc một Text Editor nào ta thích để phát triển trang ASP.NET mà thôi.

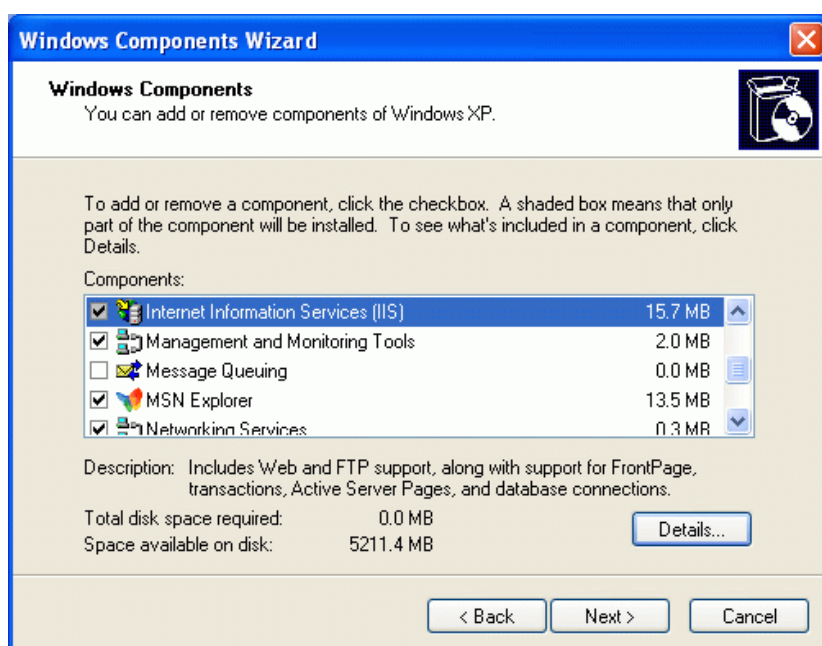
Nhớ là ASP.NET là kỹ thuật phát triển mạng ở phía Server, do đó ta phải cần có Internet Information Server (hay thường được gọi đơn giản hơn là Web Server) để soạn (phát triển hay lập trình) các trang về mạng cho khách vãng lai ghé thăm cũng như tham khảo các thông tin liên hệ. Nhưng khác với các trang ASP cổ điển, Web Server sẽ không hiểu các trang ASP.NET nếu như ta quên hay bỏ sót không cài .NET Framework SDK hoặc không cài MS Visual Studio.NET, chính nhờ ở .NET Framework SDK mà ta có đầy đủ các công dụng và các object hay classes cần thiết cho các trang ASP.NET của ta.

CÀI INTERNET INFORMATION SERVER (IIS)

Internet Information Server (IIS) Version 5.0 là một sản phẩm 'cho không ... biếu không' của Microsoft khi ta mua MS Windows 2000 Professional hay MS Windows XP Professional. IIS chuyên trị về Web Server, qua đó ta có thể cung cấp các dịch vụ nói chung về mạng cho khách vãng lai, mà dịch vụ về mạng thì thiên hình vạn trạng, nhỏ xíu như từ cây kim sợi chỉ cho đến vĩ đại như phi thuyền, giải Ngân Hà hay vũ trụ, đều có thể được bố trí đầy đủ thông tin hay các ứng dụng liên hệ cần thiết để đáp ứng nhu cầu của khách vãng lai.

1. Để cài IIS Version 5.0 trong MS Windows XP Professional, ta bắt đầu chọn:

- Start, Settings, Control Panel, Add/Remove Programs và nhấp đơn (click) Add/Remove Windows Components, xong chọn Internet Information Server như sau:



2. Nếu ta nhấp đơn nút <Details>, ta có thể tự do lựa chọn thêm hay bớt các thành phần trong IIS, tỷ như ta có thể bố trí thêm File Transfer Protocol Service (FTP Server) để quản lý một cách hiệu quả hơn việc tải lên (upload) hay tải xuống (download) các hồ sơ (documents) hay tập tin (files).

3. Nhấp nút <Next>, Windows XP Professional sẽ thu thập các thông tin liên hệ và bắt đầu tiến trình cài đặt IIS. Chỉ trong vòng vài phút là ta đã có một Web Server ngon lành trong máy vi tính. Giờ đây, bạn đã trở thành một Web Master ... 'bắt đầu rồi'. Xin chúc mừng tân Web Master, ít ra ta cũng trở thành Web Master chính máy vi tính của ta (Web Master của local host).

4. Để xác định việc cài thành công Web Server, ta có thể thử như sau:

- Mở Browser của bạn, tỷ như Microsoft Internet Explorer và gõ hàng chữ như sau vào hộp địa chỉ (Uniform Resource Locator) <http://localhost> hay <http://cantho>

cantho (Cần Thơ) trong trường hợp này chính là tên máy vi tính của tôi đó (Computer Name). Ta có thể gọi **localhost** hay **cantho** tùy ý.

- Sau khi ta nhấp nút <Enter>, trang mặc định (default page) của IIS sẽ hiển thị như sau:



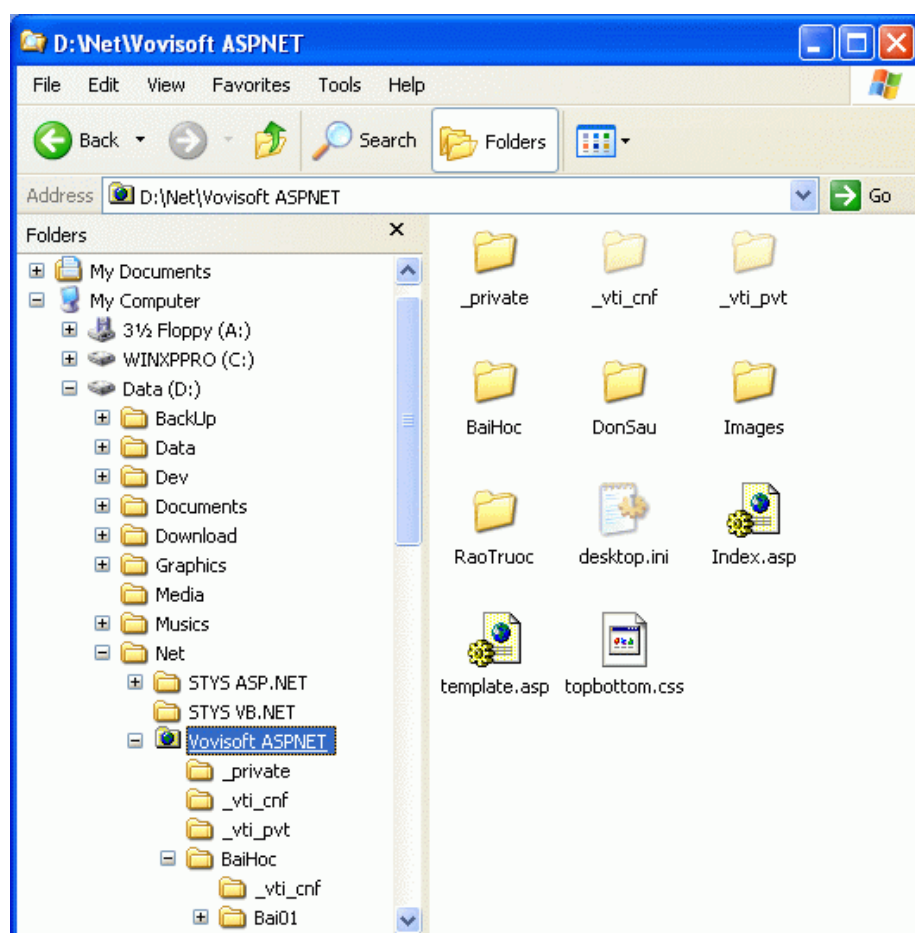
BỔ TRÍ VIRTUAL DIRECTORY DÙNG TRONG KHÓA TỰ HỌC ASP.NET CỦA VOVISOFT

Trước khi ta có thể tạo ra và bổ trí một **virtual directory** tên là **VovisoftASPNET** (hoặc bất cứ tên nào bạn thích) dùng trong khóa Tự Học ASP.NET này, ta phải làm một số việc như sau:

- Tạo ra 1 sub folder Vovisoft ASPNET cho các bài tập (Exercises).
- Vào (access) bên trong Web Server để kiểm tra hay quản lý các tài nguyên trong mạng.
- Bổ trí virtual directory VovisoftASPNET.

Chi tiết các bước theo thứ tự như sau:

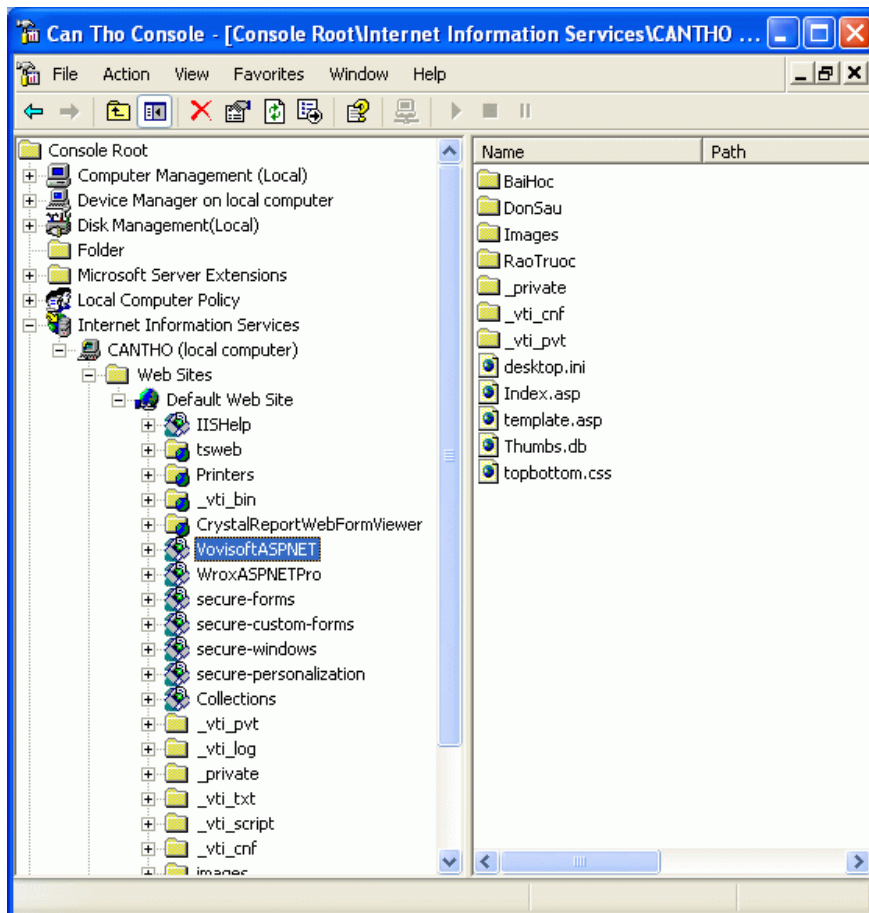
1. Tạo ra một sub directory có tên là **Vovisoft ASPNET** cấu trúc ở drive D như sau: **D:\Net\Vovisoft ASPNET**, để ý là có một chỗ trống giữa 2 chữ Vovisoft và ASPNET. Tuy vậy, nếu bạn nhuần về IIS, bạn có thể đặt tên tùy thích và không nhất thiết phải theo y chang kiểu mẫu này, nhưng bạn phải hiểu việc bạn làm để bố trí một virtual directory thích hợp dùng thực hành các bài tập trong khóa sao cho thành công.



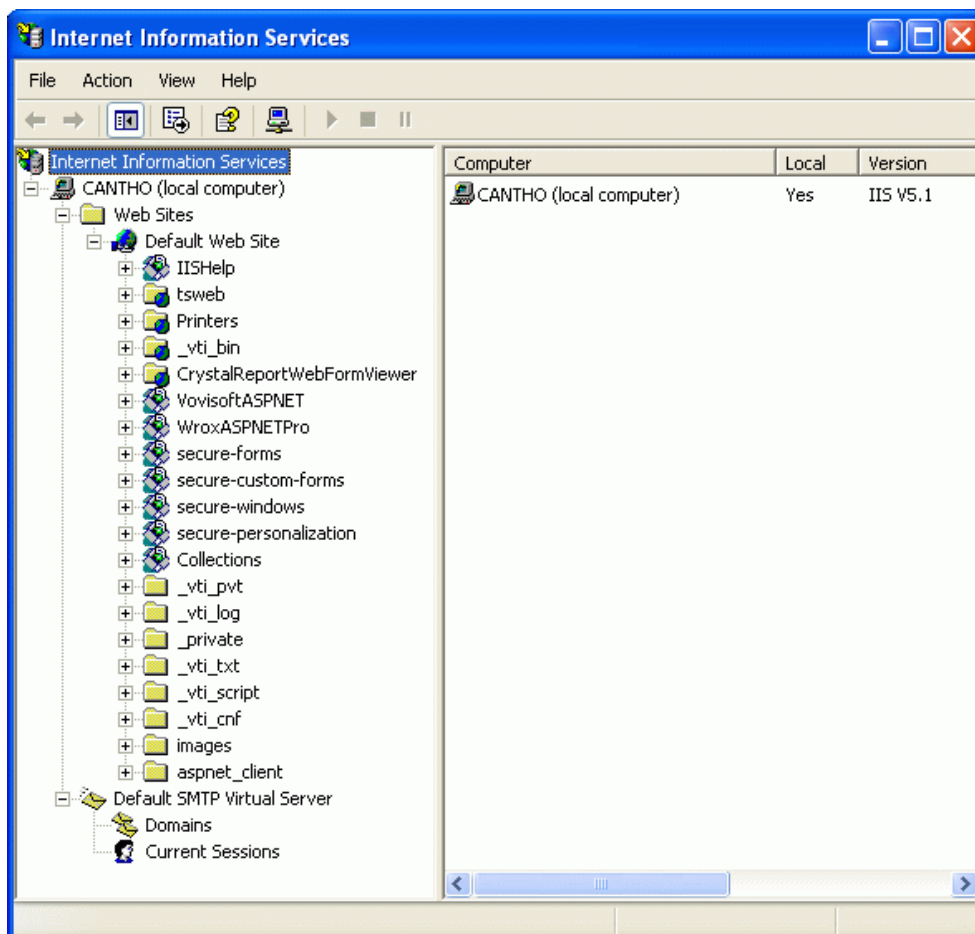
2. Có nhiều phương pháp vào (access) bên trong Web Server trước khi bố trí virtual directory, ta có thể dùng **MMC (Microsoft Management Console)** để cộng thêm **IIS snap-in** vào console hoặc ta có thể chạy IIS bằng cách chọn: **Start, Programs, Administrative Tools, Internet Information Service**.

Phương pháp dùng MMC là phương pháp được các **MCP (Microsoft Certified Professional)** hay **MCSE (Microsoft Certified System Engineer)** ưa thích hơn vì tính cách linh động mềm dẻo thích hợp cho việc quản lý các nguồn tài nguyên khác nhau trong mạng.

Dưới đây là hình đặc trưng cho việc dùng MMC:

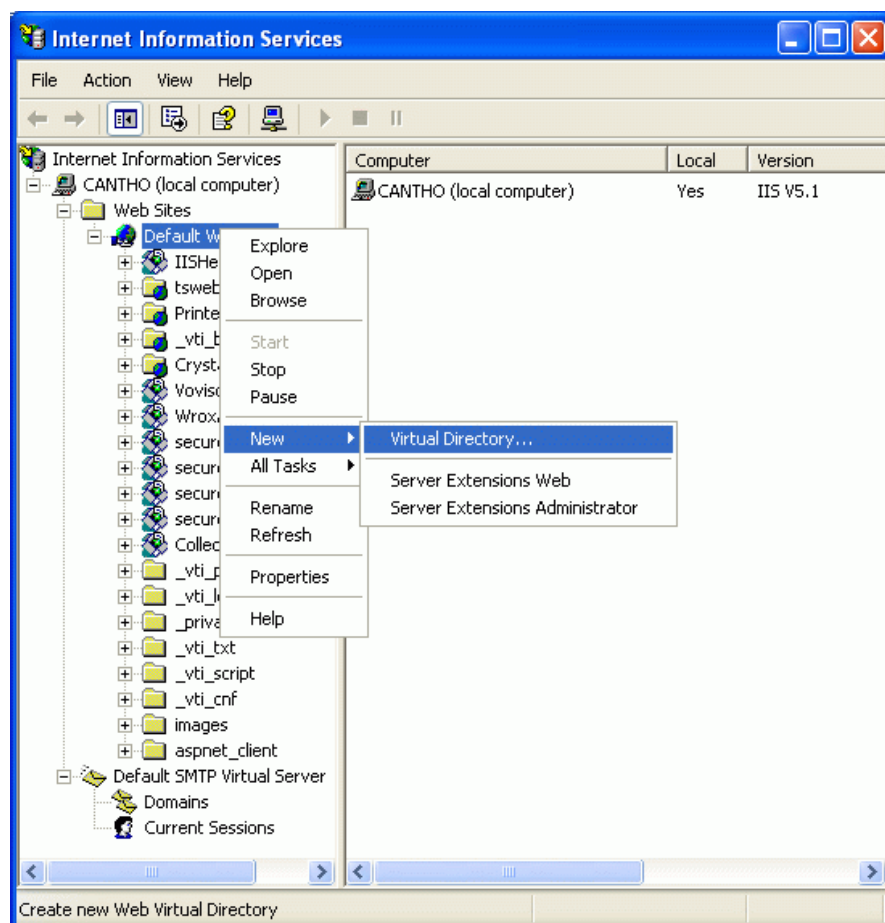


Tuy nhiên, ta sẽ không đào sâu vào chi tiết việc xử dụng MMC để bố trí virtual directory cho khóa học ở đây mà ta sẽ dùng kiểu vào IIS trực tiếp qua việc chọn: **Start, Programs, Administrative Tools, Internet Information Service** như đã trình bày ở trên. IIS sẽ hiển thị như sau:



3. Bố trí một virtual directory VovisoftASPNET:

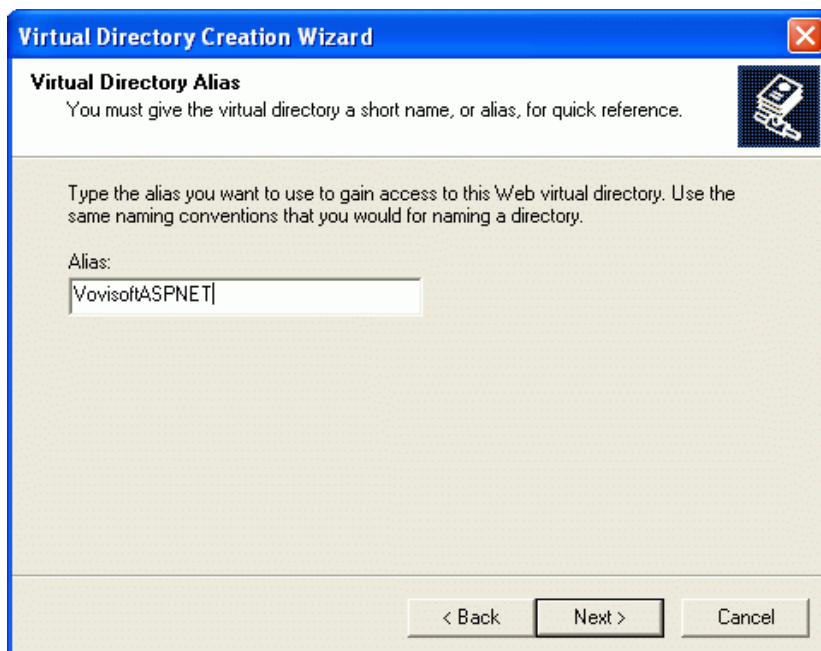
- Nhấp phải (right click) vào Default Web Site, chọn New, Virtual Directory:



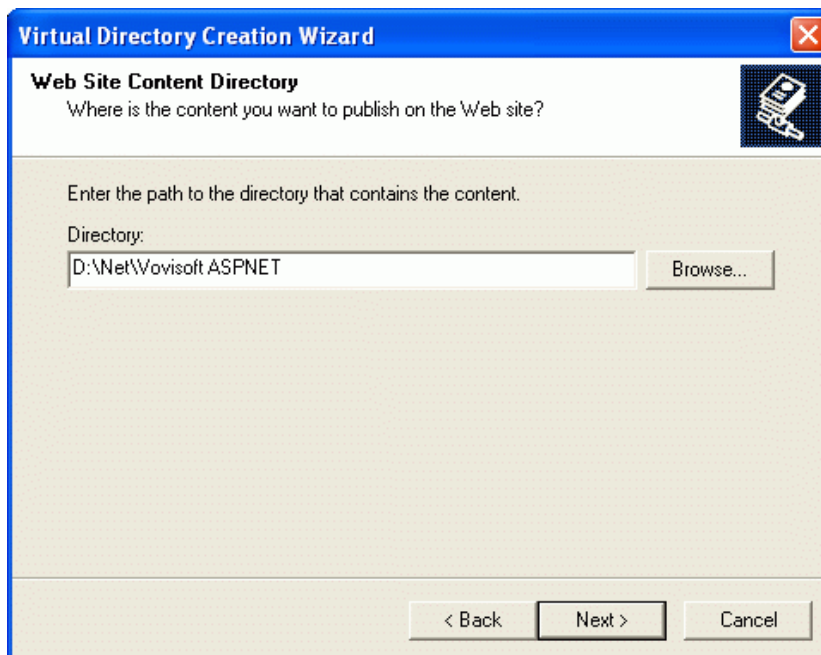
- Virtual Directory Creation Wizard sẽ bắt đầu tiến trình bố trí:



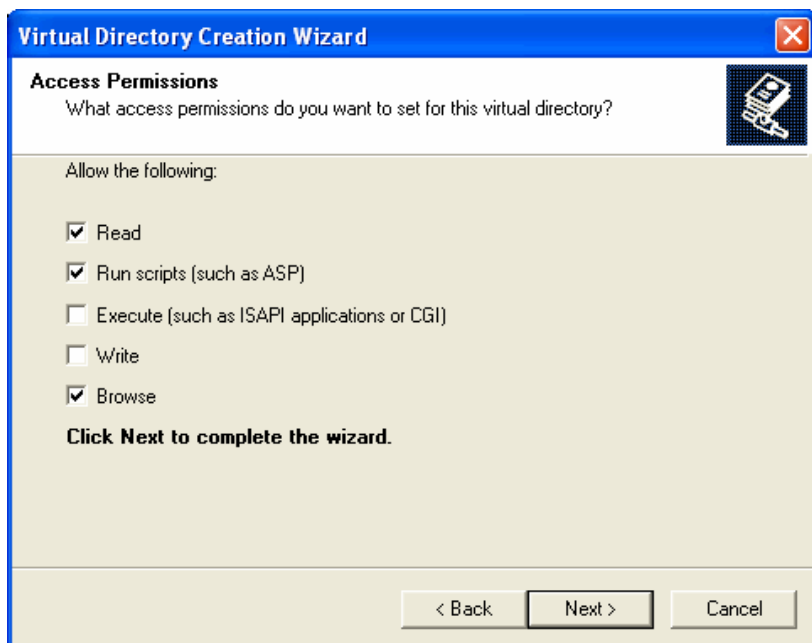
- Nhấp vào nút <Next> và gõ **VovisoftASPNET** ở hộp **Alias**:



- Nhấp vào nút <Next> và dùng <Browse...> để chọn sub folder 'D:\Net\Vovisoft ASPNET' mà ta đã chuẩn bị trước đây cho các bài tập của khóa.



- Chấp nhận các bố trí mặc định (default settings) cho virtual directory VovisoftASPNET. Nếu thích, ta có thể thêm vào đó quyền Browse (mục đích là sau này ta dễ truy cập và chạy các trang ASP.NET phát triển cho ứng dụng của mình).
- Nhấp <Next> và sau đó <Finish> để kết thúc tiến trình bố trí.



Thế là ta đã có một virtual directory để chạy các trang ASP.NET sau này. Bạn thấy đó! Việc tạo ra 1 virtual directory còn dễ dàng hơn là ăn ... 'tô tái nạm'.

CÀI MS VISUAL STUDIO.NET

Xin tham khảo [bài học số 1](#) của [thầy Lê Đức Hồng](#) ở khóa **Học Microsoft .NET Framework** về việc làm sao để cài đặt MS Visual Studio.NET thành công.

Lưu ý: Khi cài MS Visual Studio.NET là ta cũng đã cài .NET Framework SDK, do đó ta không cần phải theo những chỉ dẫn về việc cài đặt .NET Framework SDK như trình bày ở bước kế tiếp.

CÀI .NET FRAMEWORK SOFTWARE DEVELOPMENT KIT (SDK)

Tuy ta có thể tải xuống miễn phí .NET Framework SDK ở mạng www.microsoft.com/.NET nhưng lưu ý là SDK này gồm tổng cộng gần hơn 130 MBytes nên ta sẽ mất rất nhiều thời gian (khoảng 6, 7 giờ) để tải thành công trong trường hợp dùng 56 Kbps modem. Ngoài ra, thay vì tải qua mạng, ta cũng có thể đặt mua SDK chứa trong CD-ROM hay truy cập SDK trong các CD-ROM biếu kèm với các tạp chí Tin Học.

Một khi đã có .NET Framework SDK rồi, ta chỉ cần chạy chương trình **setup** là xong. Tiến trình này cũng sẽ rất lâu vì Windows sẽ khai mở từng tập tin một cũng như thu thập các thông tin cần thiết cho việc bố trí .NET Framework ở máy của ta, do đó tốt hơn hết

là ta ... tà tà tự pha cho mình một ly cà phê ... rồi vừa nhâm nhi vừa thưởng thức bài học số 1 của **thầy Lê Đức Hồng** như đã nêu trong phần 'Cài MS Visual Studio.NET' để tìm hiểu thêm về .NET Framework.

TẠO TRANG ASP.NET ĐẦU TIÊN

Mặc dầu trang ASP.NET thuần túy chỉ chứa đựng chữ và ... chữ (pure text) như trang về **HTML** nhưng đầu ai cấm ta dùng MS Visual Studio.NET để soạn các trang ASP.NET và qua đó xử dụng giao diện bằng hình (**Graphic User Interface - GUI**) để việc bố trí hay phát triển thêm phần dễ dàng và linh động.

Lưu ý là phần nối thêm (extension) ở trang ASP.NET sẽ mang tên **.aspx** để phân biệt với **.asp** trong ASP cổ điển. Ta sẽ tạo trang ASP.NET đầu tiên bằng 2 phương pháp như sau:

- Dùng MS Visual Studio.NET để tạo trang ASP.NET đầu tiên.
- Dùng Notepad để tạo trang ASP.NET đầu tiên.

Bài tập 1: Dùng MS Visual Studio.NET để tạo trang ASP.NET đầu tiên.

Mục đích:

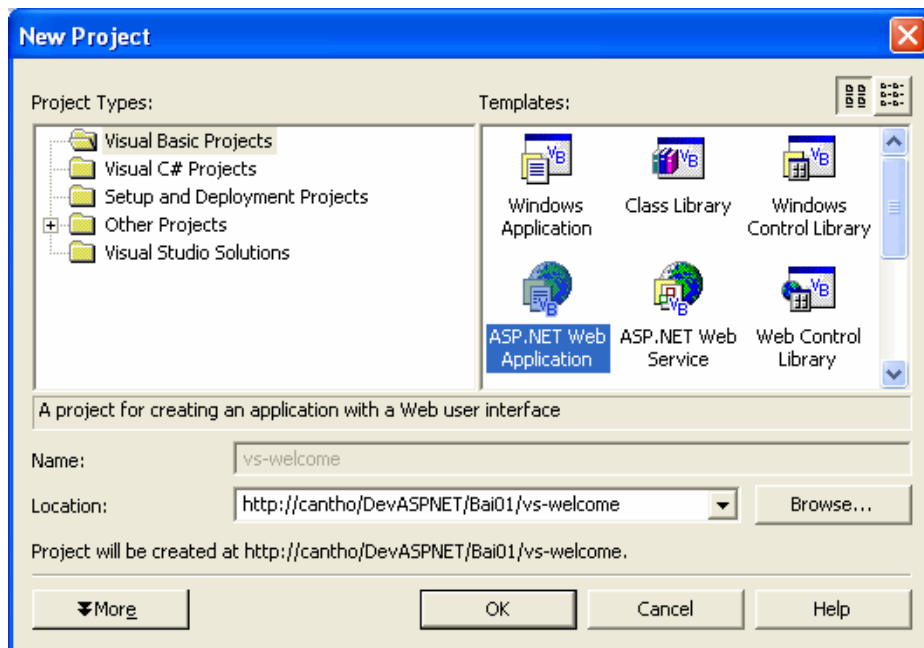
Hiện thị hàng chữ "Welcome to Khóa Tự Học ASP.NET" trong trang ASP.NET đầu tiên để chào mừng các bạn đến với khóa học này của Vovisoft.

Các bước thứ tự như sau:

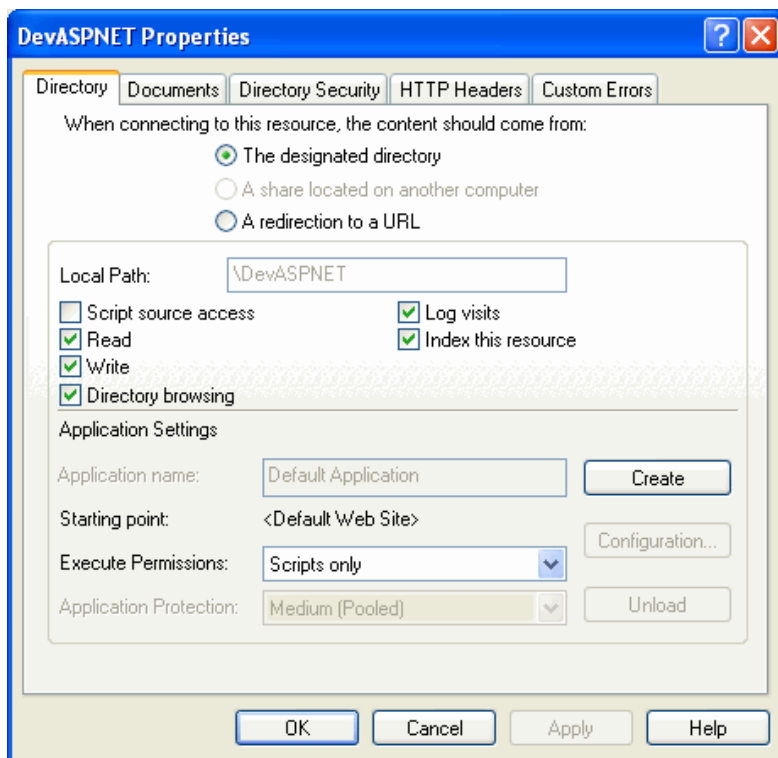
1. Chạy MS Visual Studio.NET và chọn dự án mới (New Project) như sau:

Project Types:	Visual Basic Projects
Templates:	ASP.NET Web Application
Name:	vs-welcome
Location:	http://cantho/DevASPNET/Bai01/

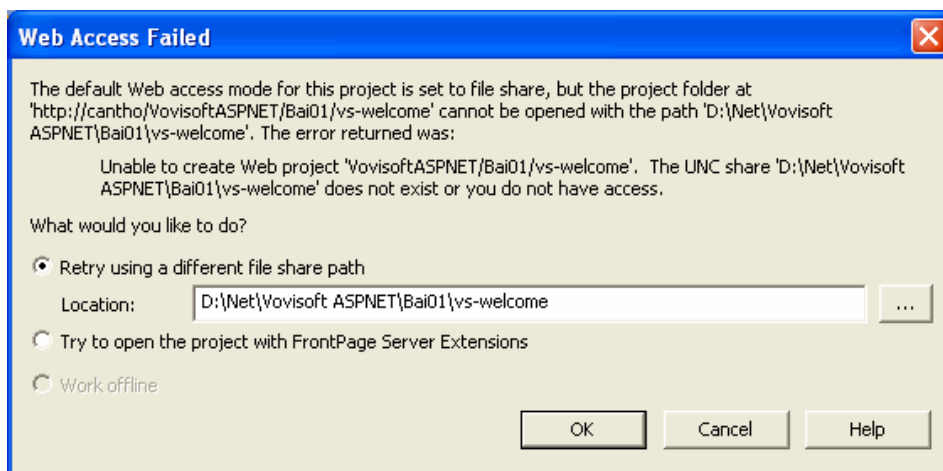
Ta nên nhớ kiểm tra hàng chữ phía dưới hộp Location để coi lại xem dự án được tạo ra ở đúng chỗ mà ta muốn chứa hay là không? Chính hàng này cũng là URL (Uniform Resource Locator) ta cần phải điền vào ở hộp địa chỉ (Address) trong Internet Explorer để hiển thị trang ASP.NET với phần nối thêm (extension) là **.aspx**



MS VS.NET sẽ tạo các sub-folders như sau: DevASPNET\Bai01\vs-welcome ở bên dưới Default Web Sit, trong trường hợp này - Default Web Site chính là **C:\inetpub\wwwroot**. Nếu dùng IIS để kiểm tra đặc tính của DevASPNET, ta thấy Local Path là DevASPNET.



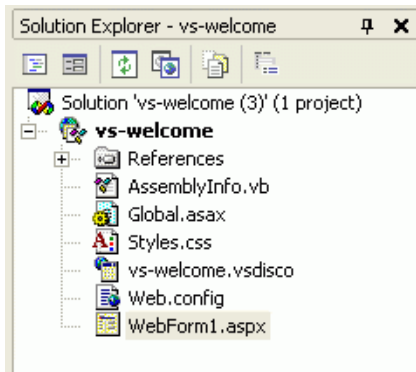
Bạn có thể chọn Location nào tùy ý theo sự tổ chức lưu trữ các dự án của bạn, nhưng nhớ đừng trùng với tên của các Virtual Directory đã có sẵn. Tỷ như ta đã tạo trước 1 virtual directory tên là VovisoftASPNET, sau đó dùng MS VS.NET để tạo 1 ASP.NET Web Application ở Location <http://cantho/VovisoftASPNET/Bai01/vs-welcome>, ta sẽ gặp lỗi sau đây khi Web Server không cho phép ta 'overwrite' virtual directory đó:



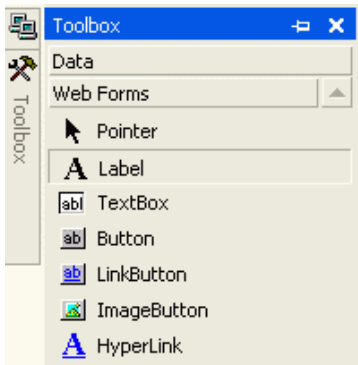
Lưu ý:

Như đã trình bày, chúng ta sẽ không đào sâu chi tiết về cách sử dụng MS VS.NET ở đây, xin tham khảo ở các bài Học .NET Framework do thầy Lê Đức Hồng soạn.

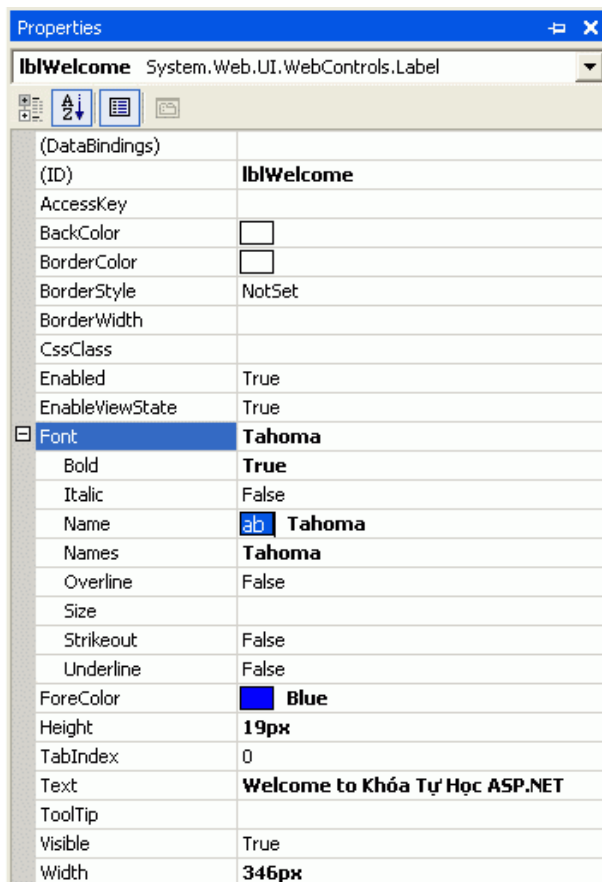
2. Nhấp vào nút <OK> để MS VS.NET tạo ra 1 dự án mới. Nếu để ý, ta sẽ thấy một trang ASP.NET được tạo ra một cách mặc định (default) với tên là WebForm1.aspx nằm trong Solution Explorer Windows về mé tay phải, bên trái ta có 1 hộp dụng cụ (Toolbox) và ở giữa là View Designer.



3. Mở hộp dụng cụ (Toolbox) và nhấp đôi vào công cụ Label, một cách thông dụng khác là ta có thể kéo lê (**drag**) công cụ Label vào trong View Designer.



4. Bố trí các đặc tính của công cụ Label đó như sau:



Lưu ý ở đây là ta sẽ dùng ứng dụng **VPSKEY Version 4.0 (or up)** với **bảng mã Unicode** để gõ tiếng Việt cho đặc tính Text của Label tên **lblWelcome** như hình trên. Ta có thể dùng bất cứ ứng dụng gõ tiếng Việt nào cho phép encoding với Unicode, nếu dùng VPSKeys Version 4.0 thì bạn cần phải bố trí như sau:



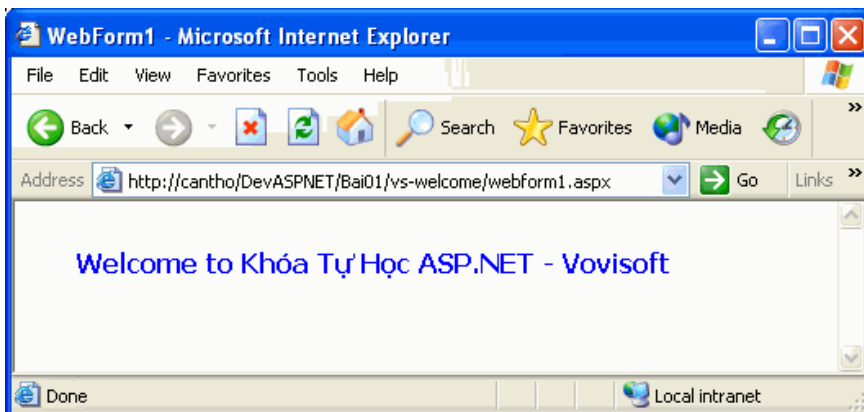
5. Chọn File, Save WebForm1.aspx As để lưu trữ vào đĩa cứng, nhớ chọn Save with Encoding (hay ta có thể dùng Advanced Save Options) với Encoding là **Unicode (UTF-8 with signature) - Codepage 65001** để lưu trữ (save) tiếng Việt chính xác.



6. Cuối cùng ta sẽ dùng IE Client Browser để hiển thị trang **WebForm1.aspx** (nếu ta không đổi tên, trang mặc định là trang WebForm1.aspx khi dùng MS VS.NET để tạo 1 ASP.NET Web Application).

Trong hộp địa chỉ (Address), lưu ý là:

- <http://cantho/DevASPNET/Bai01/vs-welcome> được MS VS.NET tạo ra trong Server **cantho** (hay có thể dùng **localhost**)
- **webform1.aspx** là trang web mà ta vừa phát triển



Bài tập 2: Dùng Notepad để tạo trang ASP.NET đầu tiên.

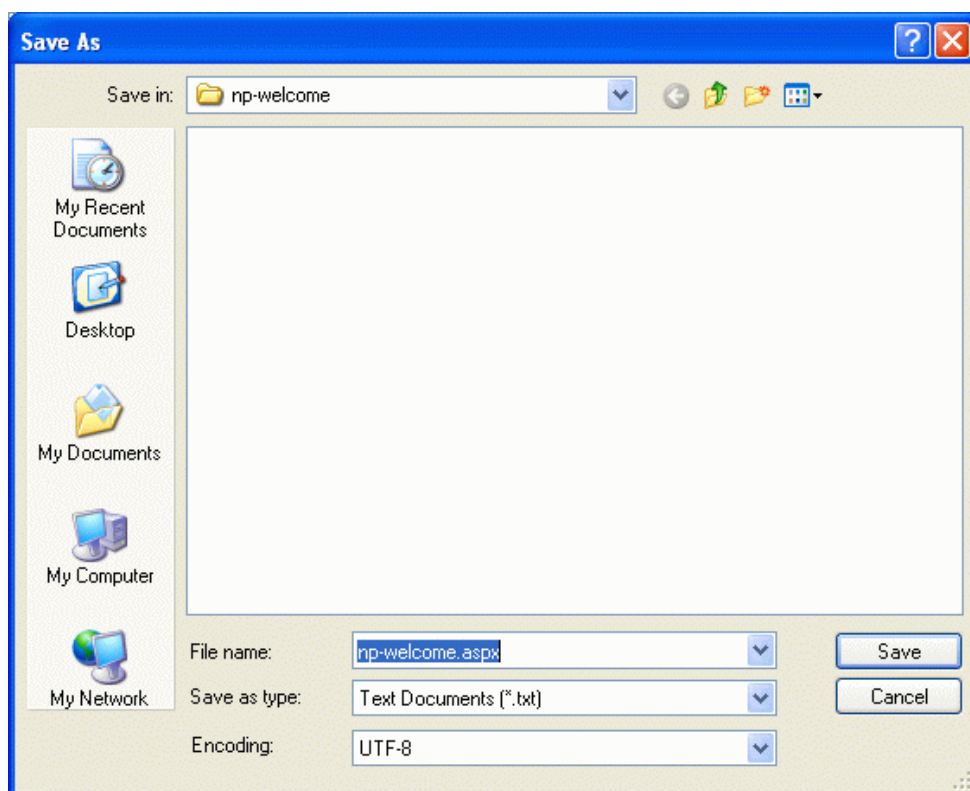
Mục đích:

Ở bài tập 2, ta cũng hiển thị hàng chữ "Welcome to Khóa Tự Học ASP.NET" trong trang ASP.NET để chào mừng các bạn đến với khóa học này của Vovisoft. Tuy nhiên, ở bài tập này, ta chỉ dùng **Notepad** để tạo ra 1 trang ASP.NET y chang như trang ở trên.

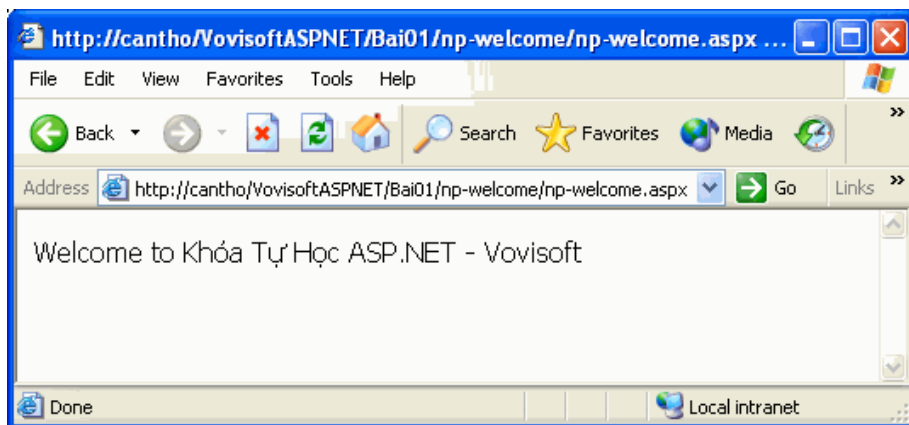
1. Chạy ứng dụng Notepad và gõ những hàng chữ lập trình y chang như trong hình dưới đây. Trong bài học 1, ta tạm thời chấp nhận các mệnh lệnh và cú pháp trong trang ASP.NET đầu tiên này và sẽ tham khảo chi tiết cũng như phân tích và giải thích về phương pháp xây dựng trang ASP.NET ở bài học kế tiếp. Nhớ khi gõ tiếng Việt ta sẽ dùng VPSKeys Version 4.0 như trong bài tập 1.

2. Đặt tên tập tin (file) này là **np-welcome.aspx** và lưu trữ (save as) với encoding UTF-8 (UTF-8 là chữ viết tắt của **Universal Transform Format-8**) dưới subdirectory **D:\Net\Vovisoft ASPNET\Bai01\np-welcome**

Khi lưu trữ np-welcome.aspx dưới 1 subdirectory như trên, ta cố tình muốn dùng **virtual directory VovisoftASPNET** đã tạo ra trong IIS để hiển thị trang web này.



3. Hiển thị trang ASP.NET đầu tiên này với IE Client Browser và gõ hàng chữ dưới đây vào hộp địa chỉ (Address): **http://cantho/VovisoftASPNET/Bai01/np-welcome/np-welcome.aspx** rồi nhập vào nút <Enter>. Ta sẽ có 1 trang ASP.NET với hàng chữ Welcome to Khóa Tự Học ASP.NET - Vovisoft như sau:



Tóm tắt

Trong bài học này, ta thật sự làm quen với ASP.NET qua việc tìm hiểu các phương pháp làm việc của mạng trong phạm vi ASP.NET. Thêm nữa, ta cũng biết sơ lược về sự thay đổi cơ bản trong việc phát triển mạng do nhờ có .NET Framework và bắt tay vào việc cài đặt ASP.NET để thực hiện được trang ASP.NET đầu tiên.

Thật vậy, qua 2 bài tập, ta đã xử dụng thành công 2 phương pháp với 2 ứng dụng khác nhau để tạo ra trang Web '**Chào Mừng Bạn Đến Với Khóa Tự Học ASP.NET của Vovisoft**', và như ta đã thấy, mỗi cách đều có cái hay riêng, thích dùng cách này hay cách kia tùy thuộc vào thói quen, sở thích cũng như môi trường làm việc hay hoạt động xung quanh. Còn 1 phương pháp nữa mà tôi rất thích vì tính cách cộng đồng của nó là MS ASP.NET Web Matrix. Phương pháp này sẽ trình bày ở bài số 3.

Trong Bài số 2 '**Xây Dựng Trang ASP.NET**', ta sẽ tham khảo chi tiết việc xây dựng một trang ASP.NET như thế nào cũng như đào sâu về việc phân tích cú pháp và các mệnh lệnh (commands hoặc keywords) dùng trong các trang ASP.NET

Download Source Code

[Nguồn mã bài tập 1](#)

[Nguồn mã bài tập 2](#)

Bài làm ở nhà

Câu Hỏi 1: CLR là gì?

Câu Hỏi 2: Virtual Directory là gì? Và làm sao để bố trí thành công Virtual Directory?

Bài Làm 1: Phát triển 1 trang ASP.NET dùng MS Visual Studio.NET hay Notepad để hiển thị (display) 1 bài thơ hoặc các câu Ca Dao (bằng tiếng Việt) mà bạn sưu tầm.

Bài 02

Xây Dựng Trang ASP.NET

Về đây nghe em
Về đây, mặc áo the, đi guốc mộc - Kể chuyện tình - bằng lời ca dao
Kể chuyện tình - bằng nỗi ngổ khoai - Kể chuyện tình - bằng hạt lúa mới
Và về đây, nghe gọi tiếng xưa - Để nhớ trong tiếng vỗ bờ.
(Về Đây Nghe Em - Thơ A Khuê - Nhạc Trần Quang Lộc)

Trong Bài 'Làm Quen Với ASP.NET', ta nhận thức được ASP.NET giới thiệu một phương pháp mới về lập trình để phát triển trang web theo kiểu mẫu event-driven giống như dùng ngôn ngữ lập trình Visual Basic. Những ai từng làm quen và chỉ lập trình với ngôn ngữ script (script language) tỷ như VBScript hay JavaScript sẽ ngỡ ngàng đôi chút vì mã trong các trang ASP.NET không còn trộn lẫn với HTML nữa, nhằm mục đích cung cấp kiểu phát triển mới đơn giản hơn, cấp tiến hơn và cấu trúc chặt chẽ hơn (more advantages and more structure code).

Hôm nay, ta lại 'về đây' trong bài tham khảo về phương pháp xây dựng trang ASP.NET (hay gọi là Web Form) và một số vấn đề liên hệ như sau:

- Phân tích mã ở trang ASP.NET đầu tiên
- Xây dựng một trang ASP.NET đơn giản
- Vài nhận xét khi dùng ASP.NET và HTML

PHÂN TÍCH MÃ Ở TRANG ASP.NET ĐẦU TIÊN

Ta thử xem lại nguồn mã Bài tập 2 của trang ASP.NET đầu tiên trong Bài 01 'Làm Quen Với ASP.NET':

```
<%@ Page Language="VB" %>

<script runat="server">
sub Page_Load(obj as object, e as eventargs)
lblWelcome.Text = "Welcome to Khóa Tự Học ASP.NET - Vovisoft"
end sub
</script>

<html>
<body>
<asp:Label id="lblWelcome" runat="server"/>
</body>
</html>
```

Ta nhận thấy mã của trang này được chia ra làm 3 phần riêng biệt:

Phần 1: **<%@ Page Language="VB" %>** được gọi là **Page Directives**: phần này cung cấp cho ASP.NET những thông tin đặc biệt để ASP.NET biết cách mà đối xử cũng như những thông tin dùng trong tiến trình biên dịch (during the compiling process), trong đó, ta muốn ASP.NET dùng VB.NET làm ngôn ngữ lập trình mặc định (default programming language) cho trang web, ở các đề tài sau, ta sẽ tham khảo thêm về **import** directive.

Phần 2: **<script runat="server"> ... </script>** : phần này còn gọi là **Code Declaration Block** giống như mã ở Client Side nhưng khác một chút là có kèm theo **runat="server"** chỉ thị cho ASP.NET biết thi hành trang này ở Server Side, phần này cũng là đặt ra ... 'dụng võ', kiểm soát mọi công dụng cần thiết và mã **được biên dịch (compiled) thành MSIL**. Thật ra, ta có thể đặt phần này ở bất cứ nơi nào trong trang web nhưng nếu sắp xếp ở phần đầu tiên của trang cũng là một thói quen tốt để phân biệt mã của ASP.NET với mã của HTML.

```
sub Page_Load(obj as object, e as eventargs)
lblWelcome.Text = "Welcome to Khóa Tự Học ASP.NET - Vovisoft"
end sub
```

Phần mã này tạo ra một Procedure có tên là Page_Load mặc định (default) cho các trang ASP.NET giống như Form_Load trong việc lập trình dùng ngôn ngữ Visual Basic với sự cố (event) **Load**, khi trang đầu tiên này được đưa ra trình bày, ASP.NET sẽ gắn hàng chữ "**Welcome to Khóa Tự Học ASP.NET - Vovisoft**" vào hộp chữ có nhãn hiệu (label) **lblWelcome**.

Phần 3: **<html>...</html>** : đây là nơi ta bắt đầu phần mã của HTML. Phần này chính là hình thức trình bày nội dung của trang được chế biến bởi mã ASP.NET trước khi gọi về và hiển thị trong browser của Client. Ngoài ra, ASP.NET cũng cho phép ta kèm theo những chỉ thị (instructions) trong **Code Render Block** bắt đầu với **<%** và chấm dứt với **%>** tỷ như:

```
<% Response.Wite ("My first page <P>") %>
```

để đưa ra những gì ta muốn làm vào bên trong lòng nguồn mã của phần HTML. Thí dụ ở đây, ta chỉ đơn giản hiển thị hàng chữ "My First Page" khi gọi trang này về browser của Client.

Phân định Mã và Nội Dung

Tiện đây, ta sẽ tìm hiểu thêm về sự phân định giữa **Mã (Code)** và **Nội Dung (Content)** gọi là **Code and Content Separation**.

Như ta đã biết, phần đông các Kỹ Sư Tin Học khi hình thành một web site đã xây dựng mọi chuyện từ A tới Z cho mạng của mình kể cả việc hoạch định các đề án thiết kế cũng như trang trí, sắp xếp và trình bày các thông tin trong các trang web. Tóm lại, các Kỹ Sư Tin Học đó không những phải chuẩn bị và viết các nguồn mã cần thiết cho các trang của mình mà có thể còn 'thầu' hoặc 'bao' luôn công việc của một Thiết Kế Gia (Designer), một Trang Trí Viên (Decorator), một Kiến Trúc Sư (Architect), một Họa Sĩ (Painter), một Giáo Sư (Instructor or Tutor), một Web Master, một Thông Tấn Viên ... và rất nhiều vai trò khác không thể nào kể xiết tùy theo mục đích và tôn chỉ của web site.

Điều này thì ... cũng tốt thôi, nhưng ta thiết nghĩ, không phải Kỹ Sư Tin Học nào cũng được trang bị ... 'thập bát ban võ nghệ' như vậy và thường tình, một Kỹ Sư Tin Học chuyên nghiệp lại dở (unskillful) về trang trí thiết kế, lý do đơn giản là họ đã tập trung khả năng, thời gian có được vào việc lập trình và xem nhẹ việc trang trí thiết kế cho web site của mình. Nhất là trong phạm vi các đại công ty hay các bộ sở chính phủ, việc xây dựng web site chứa nhiều thông tin là việc của một nhóm chứ không phải của một cá nhân, mỗi người phụ trách một việc, tỷ như người lo về mã (code), người lo về cách thức trình bày (with HTML, ...) do đó việc phân định rõ ràng giữa mã (code) và nội dung (content) trở nên rất cần thiết trong việc phát triển các trang web.

ASP.NET giải quyết vấn đề bằng 2 hướng như sau:

- Kiểu mẫu Code Inline (Code Inline Model)
- Kiểu mẫu Code Behind (Code Behind Model)

Code Inline Model:

Trong kiểu mẫu này, mã vẫn được viết và giữ (code section) trong các trang ASP.NET nhưng không trộn lẫn với HTML dành cho phần nội dung (content section) như mã ta được thấy trong bài tập 2 trong Bài 01 'Làm Quen Với ASP.NET' với 2 phần mã và nội dung nằm riêng biệt nhau:

```
<%-- This is the code section (ASP.NET Code) --%>
<script runat="server">
sub Page_Load(obj as object, e as eventargs)
lblWelcome.Text = "Welcome to Khóa Tự Học ASP.NET - Vovisoft"
end sub
</script>

<%-- This is the content section (HTML Code) --%>
<html>
<body>
<asp:Label id="lblWelcome" runat="server"/>
</body>
</html>
```

Code Behind Model:

Một kiểu mẫu khác được áp dụng để phân định mã và nội dung là phần mã được sắp xếp trong một tập tin khác riêng biệt ở một chỗ lưu trữ khác hẳn chỗ chứa các trang ASP.NET, dĩ nhiên khi ta phân chia như vậy, ta phải sắp xếp lại để 2 tập tin riêng biệt đó có thể làm việc với nhau và được biên dịch cũng giống như kiểu mẫu Code Inline. Ta sẽ đi sâu vào chi tiết kiểu mẫu này ở những bài kế khi bàn về cách **'Import'** các mã từ nguồn tài nguyên ở bên ngoài.

Tiến Trình Xử Lý

Bây giờ, ta sẽ tìm hiểu về tiến trình xử lý trang web như thế nào khi có khách vắng lai viếng thăm (hoặc lướt mạng) trang ASP.NET của ta.

Khi có người yêu cầu tham khảo trang web của ta, trước hết ASP.NET sẽ **biên dịch (compile) mã** ở phần Code Decalration Block trong trang ASP.NET. Nếu để ý một chút, ta nhận thấy có một sự trì hoãn mặc dầu nhẹ nhàng hay không đáng kể trước khi browser hiển thị trang web đó, nhưng nếu cũng chính trang đó được viếng thăm lần thứ nhì hoặc những lần sau đó thì sự trì hoãn ta nhận thấy trước kia sẽ biến mất, lý do là vì trang đó đã được biên dịch rồi, ASP.NET chỉ có việc dùng mã thôi chứ không cần phải biên dịch lần nữa, do đó hiệu suất gia tăng hết sức ngoạn mục. Nhớ là mã được ASP.NET biên dịch thành MSIL, CLR quản lý và sau đó biên dịch thành **ngôn ngữ gốc của máy (native machine language)** trước khi trang được gửi về browser của Client. Nguồn mã (code) mà ta lập trình trong trang web được lưu trữ một nơi riêng biệt, CLR sẽ thăm chừng luôn luôn, phòng trường hợp ta thay đổi nguồn mã thì lập tức CLR sẽ biên dịch lại 1 cách tự động trang web đó mà không cần ta phải làm gì cả.

Sau khi trang web được biên dịch, ASP.NET bắt đầu tiến trình xử lý tất cả nguồn mã kể cả các sự cố (events) tý như ta gõ vài chữ trong hộp chữ (text box) hoặc nhấp (click) một nút nào đó thì bộ máy ASP.NET (ASP.NET engine) sẽ nghiên cứu, khảo sát biến cố đó để quyết định cách phản ứng và thi hành để đáp ứng lại biến cố theo kế hoạch lập trình đã quy định trước.

Tiếp theo, ASP.NET biến đổi tất cả các Server Controls trong trang web ra thành những yếu tố HTML tương đương (HTML elements) tý như biến đổi `<asp:Label>` control ra thành HTML `` để hiển thị hàng chữ "Welcome to Khóa Tự Học ASP.NET - Vovisoft" như thí dụ đang phân tích ở trên (ta sẽ thấy HTML element này khi quan sát nguồn mã bằng cách chọn View, Source lúc trang ASP.NET của Bài 01 - Bài Tập 2) hiển thị trong Client Browser. ASP.NET cũng định giá những nguồn mã của Code Render Block (nếu có) và cũng sản xuất ra những yếu tố HTML tương đương.

Cuối cùng, ASP.NET sẽ gửi kết quả chung cuộc dưới dạng HTML tới Client Browser, Client Browser chỉ nhận được trang web dưới dạng phù hợp với tiêu chuẩn của HTML, trong đó ASP.NET dấu biến đi các nguồn mã hoặc các control thuộc phạm vi Server (Server Side Code and Controls).

Như vậy, ta có thể dùng bất cứ một Web Browser nào (không nhất thiết là MSIE) cũng có thể hiển thị các trang ASP.NET, thật vậy đối với Client Browser thì trang ASP.NET cũng chỉ là một trang HTML đơn giản nhưng có phần đuôi nối thêm (.aspx) hơi lạ mà thôi.

Lưu ý, đây là một khái niệm hết sức quan trọng, ta cần hiểu cho thấu đáo, bỏ qua có thể sẽ ... 'ăn năn hối hận suốt đời'. Browser chẳng qua chỉ là một ứng dụng hết sức thụ động (dump application) với nhiệm vụ diễn dịch các yếu tố HTML mà thôi. ASP.NET nắm vững điều đó, cho nên mỗi lần ASP.NET muốn gửi tới Brower những gì, nó đều dịch ra hay biến đổi ra dạng HTML. Không những vậy, ASP.NET còn khéo léo vận dụng để HTML ngây thơ (nhưng dễ thương) đó còn thi hành những tiến trình xử lý đặc biệt (mà ta sẽ tham khảo ở những bài học kế) cho ASP.NET ở phía Server mà hoàn toàn không hay, không biết gì hết.

XÂY DỰNG MỘT TRANG ASP.NET ĐƠN GIẢN

Bài Tập 1:

Mục đích:

Trong bài tập này, ta sẽ xây dựng một trang ASP.NET dùng một hộp chữ để tiếp nhận tên của khách vắng lai và in ra (hay hiển thị) lời chào người khách đó.

Các bước thứ tự như sau:

1. Chạy ứng dụng Notepad và gõ những hàng chữ lập trình y chang như mã dưới đây và lưu trữ (Save As) trong folder **D:\Net\Vovisoft ASPNET\Bai02\baitap01\simple.aspx** với encoding UTF-8 (hay bất cứ đâu tùy theo cách xếp đặt Virtual Directory mà ta bố trí với IIS). Nếu muốn gõ tiếng Việt, ta dùng VPSKeys Version 4.0 như đã trình bày ở Bài Học 01. Trong bài học này, ta tạm thời chấp nhận các mệnh lệnh và cú pháp trong trang ASP.NET và sẽ tham khảo chi tiết các Server Controls ở bài học kế.

```
<%@ Page Language="VB" %>
```

```
<script runat="server">
```

```
Sub tbMessage_Change (Sender AS Object, E As EventArgs)
```

```
lblmessage.Text = "Chào bạn " + tbMessage.Text
```



```
End Sub
</script>
```

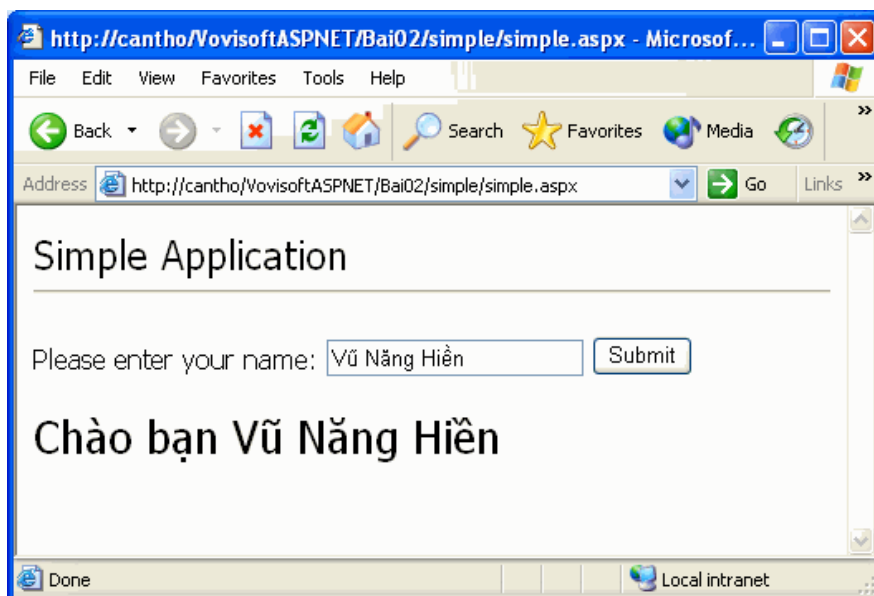
```
<html>
<body>
<font size="5">Simple Application</font><hr><p>

<form runat="server">
Please enter your name:
<asp:textbox id="tbMessage"
OnTextChanged="tbMessage_Change"
runat=server/>
<asp:button id="btSubmit" Text="Submit"
runat=server/><p>
<asp:label id="lblMessage" font-size="20pt"
runat=server/>
</form>

</body>
</html>
```

2. Hiện thị trang ASP.NET này với IE Client Browser và gõ hàng chữ dưới đây vào hộp địa chỉ (Address):

http://cantho/VovisoftASPNET/Bai02/baitap01/simple.aspx rồi nhấn nút <Enter>. Ta có thể gõ tên của mình vào hộp chữ kế bên hàng danh hiệu (label) "Please enter your name:" và nhấp vào nút Submit, ta sẽ có 1 trang ASP.NET chào mừng khách vãng lai như sau:



Phần Chú Thích:

```
<%@ Page Language="VB" %>
```

Được dùng để chỉ thị cho ASP.NET ta dùng VB.NET làm ngôn ngữ lập trình mặc định.

```
<script runat="server">
Sub tbMessage_Change (Sender AS Object, E As EventArgs)
lblmessage.Text = "Chào bạn " + tbMessage.Text
End Sub
</script>
```

Khi sự cố **tbMessage_Change** khởi động, ta dùng **lblmessage.Text** để hiển thị hàng chữ **"Chào bạn " + tbMessage.Text** ở trang ASP.NET.

```

<form runat="server">
Please enter your name:
<asp:textbox id="tbMessage"
OnTextChanged="tbMessage_Change"
runat=server/>
<asp:button id="btSubmit" Text="Submit"
runat=server/><p>
<asp:label id="lblMessage" font-size="20pt"
runat=server/>
</form>

```

Phần HTML này dùng để trình bày các Server Controls trong trang ASP.NET theo thứ tự từ trên xuống dưới, các thay đổi khác (tỷ như thay đổi do việc khởi động 1 sự cố nào đó) sẽ do phần Script chăm nom và thi hành.

Mỗi Server Control đều mang 1 ID quy ước và duy nhất, thí dụ:

<asp:textbox có ID duy nhất là **tbMessage** trong đó quy ước **tb** được phổ biến và chấp nhận riêng biệt cho công cụ **textbox**, sự cố **OnTextChanged** có tên gọi là '**tbMessage_Change**' và Server Control này <asp:textbox được thi hành ở Server Side.

Các Server Control kế gồm có <asp:button (ID **btSubmit**) và <asp:label (ID **lblMessage**) dùng cho nút bấm (ta cũng có thể dùng quy ước **btn** hay **bt** cho nút bấm) và nhãn hiệu với quy ước **lbl**.

Bài Tập 2:

Mục đích:

Trong bài tập này, ta sẽ xây dựng một trang kiểm tra vài phép toán đơn giản tỷ như phép cộng, phép trừ, phép nhân và phép chia với 2 con số nguyên. Người dùng sẽ gõ vào 2 con số nguyên và sau đó chọn phép tính bằng cách nhấn nút bấm có dấu +, -, * hay / để hiển thị kết quả.

1. Chạy ứng dụng Notepad và gõ những hàng chữ lập trình y chang như dưới đây và lưu trữ (Save As) trong folder **D:\Net\Vovisoft ASPNET\Bai02\baitap02\math.aspx** với encoding UTF-8:

```

<HTML>
<HEAD>

<script language="VB" runat="server">
Sub btAdd_Click(Sender As Object, E As EventArgs)
lblMessage.Text = "Addition Result: " & Cint(tbNumber1.Text) + Cint(tbNumber2.Text)
End Sub

Sub btSubtract_Click(Sender As Object, E As EventArgs)
lblMessage.Text = "Substraction Result: " & Cint(tbNumber1.Text) - Cint(tbNumber2.Text)
End Sub

Sub btMultiply_Click(Sender As Object, E As EventArgs)
lblMessage.Text = "Multiplication Result: " & Cint(tbNumber1.Text) * Cint(tbNumber2.Text)
End Sub

Sub btDivide_Click(Sender As Object, E As EventArgs)
lblMessage.Text = "Division Result: " & Cint(tbNumber1.Text) / Cint(tbNumber2.Text)
End Sub
</script>

</HEAD>

<BODY>

<font size="5">Simple Mathematics</font><hr><p>
<form runat="server">

Number 1: <asp:textbox id="tbNumber1" runat=server/><br>
Number 2: <asp:textbox id="tbNumber2" runat=server/><p>
<asp:button id="btAdd" Text=" + " OnClick="btAdd_Click" runat=server/>
<asp:button id="btSubtract" Text=" - " OnClick="btSubtract_Click" runat=server/>

```

```

<asp:button id="btMultiply" Text=" * " OnClick="btMultiply_Click" runat=server/>
<asp:button id="btDivide" Text=" / " OnClick="btDivide_Click" runat=server/><p>
<asp:label id="lblMessage" font-size="15pt" runat=server/>

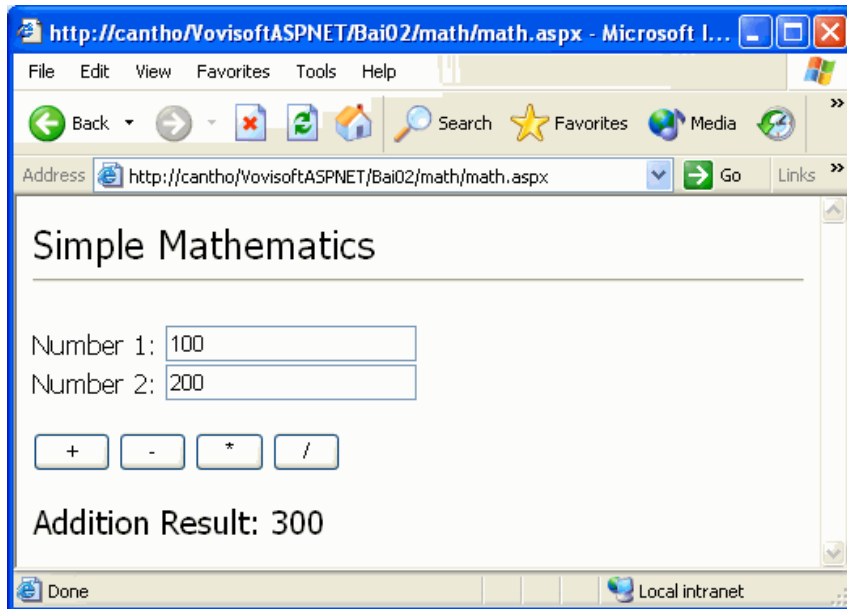
</form>

</BODY>
</HTML>

```

2. Hiện thị trang ASP.NET này với IE Client Browser và gõ hàng chữ dưới đây vào hộp địa chỉ (Address):

http://cantho/VovisoftASPNET/Bai02/baitap02/math.math.aspx rồi nhấn nút <Enter>. Trong trang kiểm tra này, ta gõ số 100, 200 vào trong các hộp chữ Number 1, Number 2 rồi nhấn nút bấm + chẳng hạn để hiện thị hàng chữ '**Addition Result: 300**'



Phần Chú Thích:

```

<script language="VB" runat="server">
Sub btAdd_Click(Sender As Object, E As EventArgs)
lblMessage.Text = "Addition Result: " & Cint(tbNumber1.Text) + Cint(tbNumber2.Text)
End Sub

...

</script>

```

Phần script này dùng để thi hành các phép toán cộng, trừ, nhân và chia các số nguyên tùy theo sự chọn lựa của user, trong đó nút có dấu (+) sẽ khởi động sự cố **btAdd_Click** và subroutine **btAdd_Click** đáp ứng bằng cách cộng 2 số nguyên đã được đưa vào ở 2 hộp chữ Number 1 và Number 2 với nhau, sau đó lưu trữ kết quả ở lblMessage.

Lưu ý ở đây, ta dùng nhóm từ 'lưu trữ kết quả' chứ không 'hiển thị kết quả' ở phần script này. Lưu trữ bằng cách ấn định **lblMessage.Text** bằng "**Addition Result:** " và kết quả phép cộng, còn phần trình bày, ta phó mặc cho mã HTML.

```

<BODY>
<font size="5">Simple Mathematics</font><hr><p>
<form runat="server">
Number 1: <asp:textbox id="tbNumber1" runat=server/><br>
Number 2: <asp:textbox id="tbNumber2" runat=server/><p>
<asp:button id="btAdd" Text=" + " OnClick="btAdd_Click" runat=server/>
<asp:button id="btSubtract" Text=" - " OnClick="btSubtract_Click" runat=server/>
<asp:button id="btMultiply" Text=" * " OnClick="btMultiply_Click" runat=server/>
<asp:button id="btDivide" Text=" / " OnClick="btDivide_Click" runat=server/><p>
<asp:label id="lblMessage" font-size="15pt" runat=server/>
</form>

```

</BODY>

Phần mã HTML này chỉ dùng để trình bày các controls trong trang web mà thôi, ở đây dưới hình thức 1 form, ta bố trí 2 textbox cho Number 1 và Number 2, 4 nút bấm Cộng, Trừ, Nhân, Chia và 1 nhãn hiệu lblMessage để hiển thị kết quả phép toán đơn giản. Tất cả đều được thi hành ở Server Side.

VÀI NHẬN XÉT KHI DÙNG ASP.NET VÀ HTML

Code Declaration Blocks và Code Render Blocks

Điều làm nhứt đầu các chuyên gia mới làm quen với lập trình là không biết dùng cái nào ... vào cái nào và ... khi nào. Khi tham khảo 1 trang ASP cổ điển, ta thấy bối rối ngay vì code nằm ... 'loạn xà ngang', trộn lẫn giữa HTML code và VBScript (hay JavaScript).

ASP.NET đã đơn giản hoá được điều này. Như đã trình bày ở đầu trang, ASP.NET cung cấp 2 cách lập trình khi dùng chung với HTML code. Đó là **Code Declaration Blocks** và **Code Render Blocks**. Nhờ đó, ta có thể nhận diện ... 'cổ nhân' trong 1 trang ASP.NET dễ dàng hơn. Trong 2 cách đó, cách thứ nhất **Code Declaration Blocks** được ưa chuộng hơn vì có thể biên dịch (compile) được thành từng tập tin riêng biệt, loại bỏ cách lập trình kiểu thập cẩm (spaghetti code) và do đó trở thành phương pháp hữu hiệu nhất để phác thảo (design) các ứng dụng xây dựng các trang ASP.NET

Còn ngoài ra, tất cả đều là HTML. Ngay cả các Server Controls cũng được diễn dịch như là các HTML code đơn giản khác. Thật vậy, mặc dù các Server Controls này là những objects phía Server (server side) nhưng giao diện (interface) của chúng được HTML diễn tả ở Client Browser như HTML code. Ngay cả các event specifier cũng là HTML (về event specifier, ta sẽ tham khảo chi tiết trong các bài học sau này).

Ta cũng có thể dùng ASP.NET object để lập trình thuần tuý HTML, tỷ như:

```
<% Response.Write("Hello Vovisoft") %>
```

theo kiểu **Code Render Blocks** để hiển thị hàng chữ "Hello Vovisoft". Như vậy thì đâu có khác gì kiểu ASP cổ điển và vô hình chung, làm việc bảo trì trang ASP.NET trở nên phức tạp.

Chú thích nguồn mã (Commenting code)

Có 3 cách chú thích nguồn mã trang ASP.NET:

- Cách dùng với <!-- và -->
- Cách dùng với '
- Cách dùng với <%-- và --%>

Cách dùng với <!-- và -->

Cách này chỉ được dùng để chú thích mã HTML mà thôi.

Cách dùng với '

Trong tất cả các mã ASP.NET, ta có thể dùng kiểu chú thích tương ứng với ngôn ngữ lập trình, tỷ như C# dùng 2 dấu slashes //, ở đây, với Visual Basic.NET, dùng dấu móc đơn ' để chú thích trong phần script như sau:

```
<script language="VB" runat="server">
Sub btAdd_Click(Sender As Object, E As EventArgs)
' Tính toán và lưu trữ kết quả phép cộng
lblMessage.Text = "Addition Result: " & Cint(tbNumber1.Text) + Cint(tbNumber2.Text)
End Sub
</script>
```

Cách dùng với <%-- và --%>

Ta có thể dùng <%-- và --%> ở bất cứ đâu **nhưng không được dùng trong Code Declaration Block** để chú thích. Tuy vậy, cách này tiện ở chỗ có thể chú thích nhiều hàng một lượt, tỷ như:

```
<%-- This is the code section (ASP.NET Code)
Phần chú thích này có thể viết thành nhiều hàng --%>
<script runat="server">
sub Page_Load(obj as object, e as eventargs)
lblWelcome.Text = "Welcome to Khóa Tự Học ASP.NET - Vovisoft"
end sub
</script>
```

Viết mã thành nhiều hàng (Multiple Lines Code)

Viết mã thành nhiều hàng với HTML không thành vấn đề, tỷ như:

```
<B>Hello
Vovisoft</B>
```

tương đương với:

```
<B>Hello Vovisoft</B>
```

Nhưng với ASP.NET và VB.NET thì không đơn giản như thế. Visual Basic.NET có soạn sẵn 1 chữ (character) đặc biệt dùng trong trường hợp này là chữ **_ (underscore)** để nối nguồn mã với nhau:

```
<% Response.Write _
("Hello Vovisoft") %>
```

Lưu ý chữ **_ (underscore)** này không dùng để nối 2 hàng chữ của 1 string với nhau, tỷ như **cách viết sau đây không đúng cú pháp lập trình**:

```
<% Response.Write ("Hello _
Vovisoft") %>
```

Tóm tắt

Như vậy, ta đã tìm hiểu thêm về cấu trúc và sơ lược qua cú pháp cũng như vài mệnh lệnh, công cụ (controls) cơ bản dùng trong các trang ASP.NET, điều này sẽ giúp ta xây dựng nền tảng cho việc soạn thảo các trang ASP.NET phức tạp hơn trong tương lai. Ta sẽ tiếp tục đào sâu thêm và riêng biệt về cách xử dụng các server controls, user controls, kiểu mẫu Event-Driven cũng như tác động hỗ tương (interact) giữa ASP.NET và .NET framework.

Có 1 điều ta cần nhắc nhở, nếu bạn không có đủ các nhu liệu cần thiết như đã liệt kê ở trang Rào Trước của khoá Tự Học ASP.NET tỷ như Visual Studio.NET, MS SQL2000 Server, ... thì làm sao có thể theo học thành công khoá này, nhất là ta lại muốn xử dụng các giao diện được biên soạn sẵn giúp việc lập trình thêm thoải mái và hào hứng. Do không phải ai cũng có đủ phương tiện hay ngân sách để trang bị cho mình các nhu liệu nêu trên nên việc tự học càng thêm ... khó khăn trăm bề. 'Cái khó ... nó bỏ cái khôn' là vậy.

Nhưng ... các bạn hãy quảng gách lo đi, vì 'may mà có em, đời còn ... dễ thương'. Em ở đây là Web Matrix. **Web Matrix có cái tên chính thức là Microsoft ASP.NET Web Matrix Project**. Đó là 1 dự án được xây dựng và do nhiều Kỹ Sư Tin Học thiện nguyện (chuyên trị về .NET) đóng góp. Cái ... đã nhất là Web Matrix được cung cấp **miễn phí** (free), do đó ta xài 1 cách thoải mái và hơn nữa, Web Matrix rất nhỏ (chỉ khoảng 1.2 MBytes - so với MS Visual Studio.NET thì giống như con chuột Mickey so với con ... khủng long) nhưng rất đẹp lại uyển chuyển dễ dùng như MS Visual Studio.NET vậy.

Do đó, bài số 3 sẽ '**Giới thiệu về Web Matrix**', về sự khác biệt đối với MS Visual Studio.NET, về việc cài đặt và bố trí cũng như cách xử dụng của Web Matrix cho việc lập trình các trang ASP.NET

Download Source Code

[Nguồn mã bài tập 1](#)

[Nguồn mã bài tập 2](#)

Bài làm ở nhà

Câu hỏi 1: Làm sao phân biệt được giữa Code Declaration Block và Code Render Block?

Câu hỏi 2: Như thế nào là Code Behind Model?

Bài làm 1: Phát triển 1 trang ASP.NET dùng hiển thị cuốn lịch (Calendar).
(Lưu ý: phần giải đáp sẽ được trình bày ở bài học kế).

Bài 03

Giới thiệu về WEB MATRIX

Thò tay mà ngắt cọng ngô
Thương em đứt ruột, già dò ... ngõ lơ
Ca Dao Việt Nam

Trong bài giới thiệu về Web Matrix, ta sẽ lần lượt tìm hiểu:

- Sơ lược về Web Matrix
- Khác biệt giữa Visual Studio.NET và Web Matrix
- Các đặc điểm của Web Matrix
- Cài đặt Web Matrix
- Đi dạo một vòng với Web Matrix
- Tạo trang ASP.NET với Web Matrix

SƠ LƯỢC VỀ WEB MATRIX

Trước khi mang tên cúng cơm như hiện nay, Microsoft ASP.NET Web Matrix Project (gọi tắt là Web Matrix) khởi đầu với tên Web Studio, chuyển sang tên Mongoose, Project Saturn, Tahiti Project và cuối cùng là Microsoft ASP.NET Web Matrix.

Tên Web Matrix được chọn mang tính chất sáng tạo và thiết lập được mối liên hệ trong cộng đồng các Chuyên Gia hay Kỹ Sư Tin Học về ASP.NET. Dự án được hình thành do sự đóng góp của nhiều Kỹ Sư Tin Học thiện nguyện đã bỏ rất nhiều thời giờ riêng tư hiếm hoi (thường là các buổi tối trong ngày và cuối tuần) để mong tạo sự hợp tác chung cho cộng đồng ASP.NET chứ không chính thức khởi xướng hay tạo nên từ Microsoft. Bạn có thể tham khảo chi tiết về tiến trình hình thành Web Matrix cũng như danh tánh các Kỹ Sư Tin Học đã đóng góp vào dự án này trong nôi [Web Matrix and Project Team](#).

Web Matrix được phác thảo cho cộng đồng ASP.NET nên ta dùng miễn phí, Web Matrix lại dễ xài và rất đẹp. Tương cũng cần nói qua, một trong các mục tiêu cơ bản của dự án Web Matrix là tạo điều kiện đóng góp chung trong cộng đồng phát triển mạng nên những người khởi xướng dự án luôn khuyến khích ta tải xuống, cài đặt, xữ dụng và giới thiệu Web Matrix cũng như đóng góp ý kiến trên diễn đàn của Web Matrix để Web Matrix được hoàn chỉnh hơn.

Các Kỹ Sư Tin Học dùng ngôn ngữ lập trình C# và .NET framework để cấu tạo ra Web Matrix. Khoảng chừng 800 Classes và hơn 150,000 mã (lines of code) được dùng trong dự án này với biết bao công sức và thời giờ đã được đầu tư vào không thể nào kể xiết. Do đó, mặc dù Web Matrix rất nhỏ (ở dạng tải xuống chỉ có 1.2MB) nhưng không những Web Matrix là môi trường phát triển các trang ASP.NET về mạng một cách đơn giản và hữu hiệu, mà còn được dùng để tạo các user controls, các classes (để biên dịch thành assemblies), các dịch vụ về mạng (Web Services) và ngay cả HTML Handlers nữa. Ngoài ra, Web Matrix còn yểm trợ cấu tạo các trang HTML thuần túy, các style sheets, các lược đồ và tài liệu XML (XML schemas and documents), các SQL scripts cũng như việc định hình các tập tin về .NET (.NET configuration files such web.config and global.asax).

KHÁC BIỆT GIỮA VISUAL STUDIO.NET VÀ WEB MATRIX

Ta phải công nhận MS Visual Studio.NET là một công cụ phát triển mạng tuyệt vời. MS Visual Studio.NET được thiết kế để làm đủ mọi chuyện ... 'trên trời dưới đất' và cho mọi Chuyên Gia chứ không riêng cho những ai chuyên trị phát triển mạng. Thật vậy, MS Visual Studio.NET được dùng như là một công cụ duy nhất để tạo ra các ứng dụng về .NET, để thiết kế các dịch vụ về mạng (Web Services), phác thảo và bố trí các trang ASP.NET cho mạng và ngay cả các tập tin tạo hình tỷ như các icons hay các bitmaps, ... nhưng MS Visual Studio.NET quá lớn, lại ... không 'rẻ' và không chuyên trị về ASP.NET như các Kỹ Sư Tin Học trong nhóm ASP.NET mong muốn nên nhóm này mới hình thành công cụ Web Matrix với mục tiêu rõ rệt - rẻ, đẹp, bền và chính thức ra mắt công chúng ngày 17 tháng 6 năm 2002 mặc dù vẫn còn dưới dạng thử nghiệm (Beta testing) với tên Microsoft ASP.NET Web Martrix Project.

Sau khi cài đặt, Web Matrix chỉ vồn vện khoảng 2.5 MBytes. Bạn có thể tải [Web Marix](#) xuống ở đây. Ở dạng tải, Web Matrix chỉ có 1.2 MB vừa vận lưu trữ trong 1 đĩa cứng nhỏ (floppy). Dĩ nhiên, ta cần phải cài .NET Framework trước khi có thể dùng Web Matrix. Khi Web Matrix ra đời, Notepad đành phải lui vào bóng đêm, đương nhiên không ai còn muốn nhìn tới nữa. Thật là ... 'trời sinh ra Du (Chu Du), tại sao còn sinh ra Lượng (Gia Cát Lượng)'.

Mặc dù Web Matrix và MS Visual Studio.NET đều là những công cụ phát triển mạng rất tuyệt vời và hoàn chỉnh nhưng có vài điểm khác biệt cần lưu ý như sau:

- Web Matrix rất nhỏ và miễn phí.
- Web Matrix gồm đủ các điểm đặc trưng rất hữu hiệu như Visual Studio.NET tỷ như 'drag and drop' các Web Controls vào trong trang Web, bố trí các đặc tính (properties) và có cơ hội quan sát trang Web dưới dạng HTML, nguồn mã hay kể cả hai dạng vừa kể, cũng như có thể cộng thêm và xử dụng các công cụ từ bên ngoài để phát triển các trang web dễ dàng hơn.
- Web Matrix còn có thể yểm trợ **FTP** mà ta không thể đòi hỏi ở Visual Studio.NET
- Web Matrix có kèm theo **Matrix Web Server** (tương tự như Personal Web Server của ASP cổ điển) để dùng thử nghiệm các trang ASP.NET mà ta đang phát triển, cũng giống như ta khởi động 1 ứng dụng trực tiếp bên trong Visual Basic.NET vậy.

Tuy nhiên, mặc dù Web Matrix vượt trội Visual Studio.NET trong nhiều phương diện, Web Matrix cũng thiếu sót vài tiện nghi mà ta thích, tỷ như không yểm trợ **IntelliSense**, không yểm trợ **Debugging** các trang ASP.NET

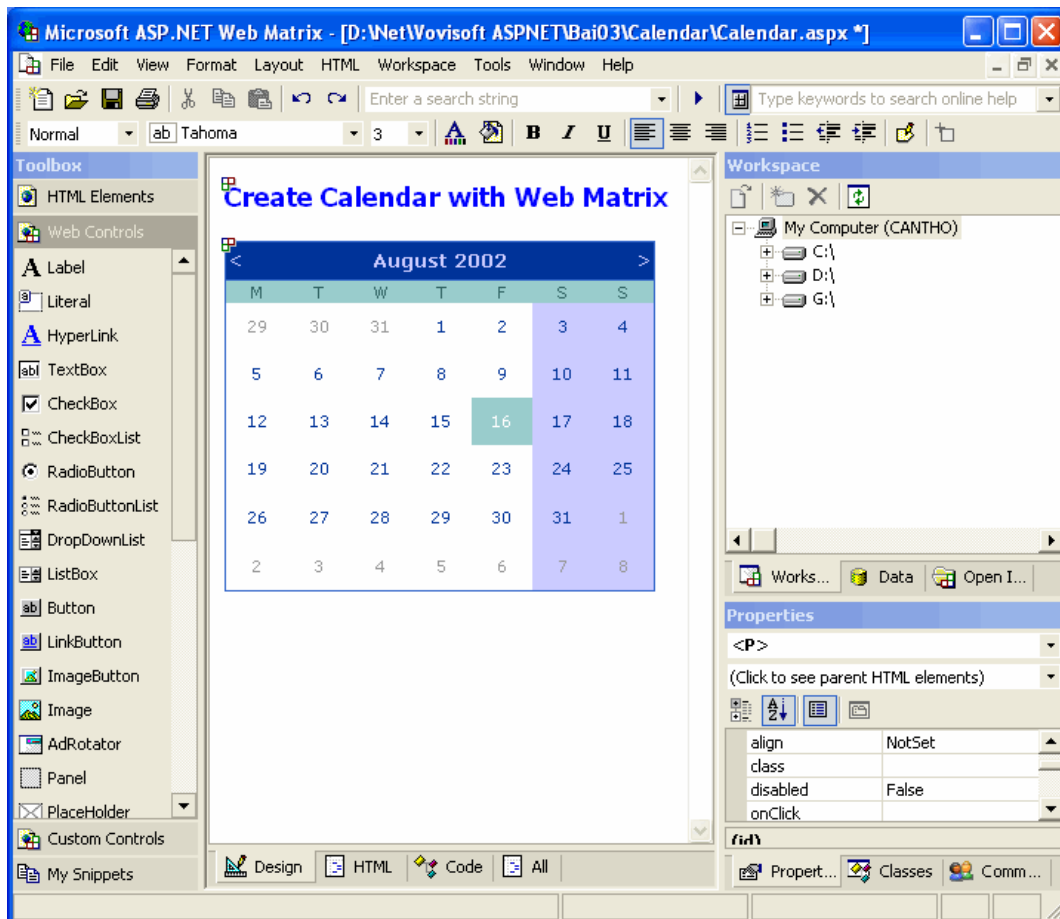
Thật sự, Web Matrix thích hợp cho những ai 'đẹp trai (hay đẹp gái), học giỏi, con nhà ... nghèo' vì 100% 'free' rất đỡ tốn. Ta thử nghĩ lại xem, Web Matrix đã đem lại nhiều hứng thú và khung IDE khá hơn nhiều so với Visual Studio.NET vì Web Matrix chuyên trị các trang ASP.NET. Còn Visual Studio.NET có thể dùng để tạo ra các thành phần (components), các công cụ đặc chế (custom controls) hay các ứng dụng về .NET (.NET applications). Tuy Web Matrix thiếu yểm trợ IntelliSense nhưng không buộc ta phải dùng code-behind development.

Ta sẽ tiếp tục tìm hiểu thêm về Web Matrix trong các phần mục kế tiếp và chắc chắn Web Matrix sẽ để lại ấn tượng sâu đậm cũng như ảnh hưởng lớn lao đến cộng đồng phát triển ASP.NET hiện tại và tương lai. Ở đây ta cũng ngã nón chào khâm phục và xin gởi một lời cảm ơn chân thành đến những người đã góp phần tạo nên một sản phẩm tuyệt vời như Web Matrix.

CÁC ĐẶC ĐIỂM CỦA WEB MATRIX

ASP.NET Page Designer

Web Matrix là một công cụ WYSIWYG (What You See Is What You Get) rất dễ xài, được tạo ra nhằm chuyên trị phát triển các trang ASP.NET. Ta có thể kéo lê (drag and drop) các ASP.NET Server Controls từ hộp công cụ (Toolbox) vào trong trang Web với đầy đủ tất cả các đặc tính (properties) cần thiết có thể sửa đổi để thích hợp mục tiêu của trang Web, nhấp đúp 1 Server Control trong trang Web sẽ tự động mở ra phần mã (Code View) để ta soạn nguồn mã cho các sự cố (events) liên kết tương ứng với các Server Event Handler.



SQL and MSDE Database Management

Web Matrix cũng kết hợp yểm trợ tạo ra, chọn lựa hay thêm bớt các hồ sơ lưu trữ trong các cơ sở dữ liệu. Việc tạo ra new databases, add, edit hay delete các tables cũng như các stored procedures và nội dung của store procedure được hình thành trực tiếp trong Web Matrix. Điều đó có nghĩa, ta không cần phải chạy Enterprise Manager (trong trường hợp dùng MS SQL 2000 Server) hay MS Access bên ngoài Web Matrix để tạo các databases hay các tables, ...

Easy Data Bound UI Generation

Web Matrix tạo điều kiện dễ dàng cho ta bố trí các **data-bound pages** mà không cần phải viết 1 dòng mã nào. Ta cũng có thể đơn giản 'drop' SQL tables vào trang Web để tạo ra các **data-bound grids** hoặc ta có thể khởi đầu với các khuôn mẫu đã soạn trước cho các báo cáo (reports) hay các trang **Master/Detail**. Ngoài ra, **Code Builders** còn giúp ta tạo ra các nguồn mã dùng để select, insert, update hay delete các SQL data.

XML Web Service Support

Web Matrix yểm trợ trình bày các dịch vụ về SOAP-based XML Web Services, cũng như gọi (call) và tiêu thụ (consume) các dịch vụ XML Web Services được cung ứng trên các Server khác.

Build Mobile Applications

Web Matrix giúp soạn nguồn mã 1 cách tự động cho các loại mobile devices khác nhau tỷ như cell phones, pagers, ... từ 1 nguồn mã chính.

CÀI ĐẶT WEB MATRIX

Trước khi cài đặt Web Matrix, ta cần kiểm tra xem các nhu liệu sau đây đã được cài đặt thành công chưa:

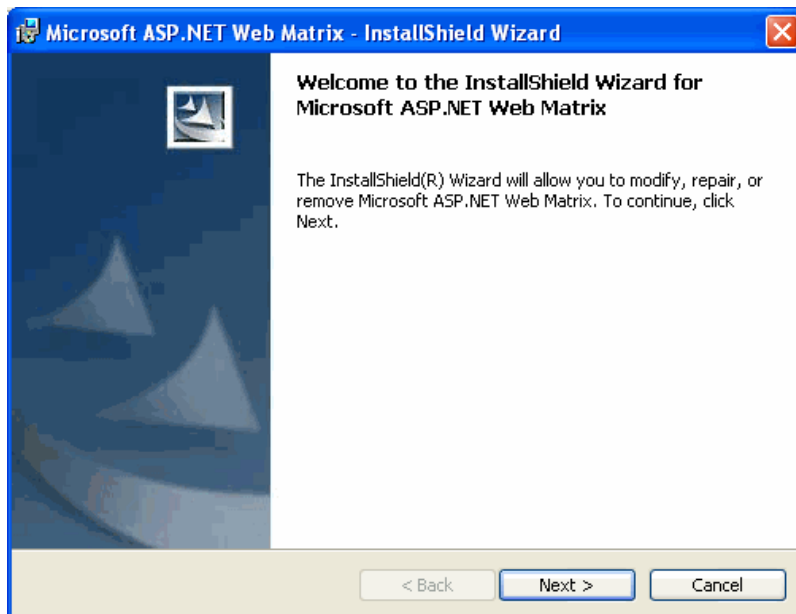
- Windows 2000 Service Pack 2 trở lên nếu ta dùng Windows 2000 Professional.

- .NET Framework.
- IE Version 5.5 trở lên.
- Windows Installer Version 2.0 trở lên.
- MS Office 2000 nếu ta muốn dùng MS Access với MSDE (MS Data Engine) làm cơ sở dữ liệu.
- MS SQL 2000 Server nếu ta muốn dùng SQL Server làm cơ sở dữ liệu.

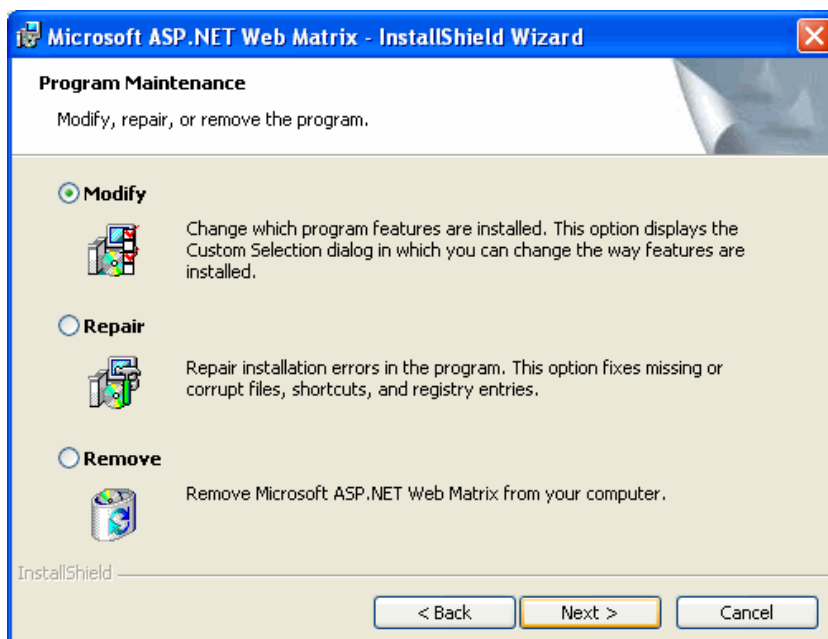
Bạn có thể tải [Web Matrix](#) xuống ở đây.

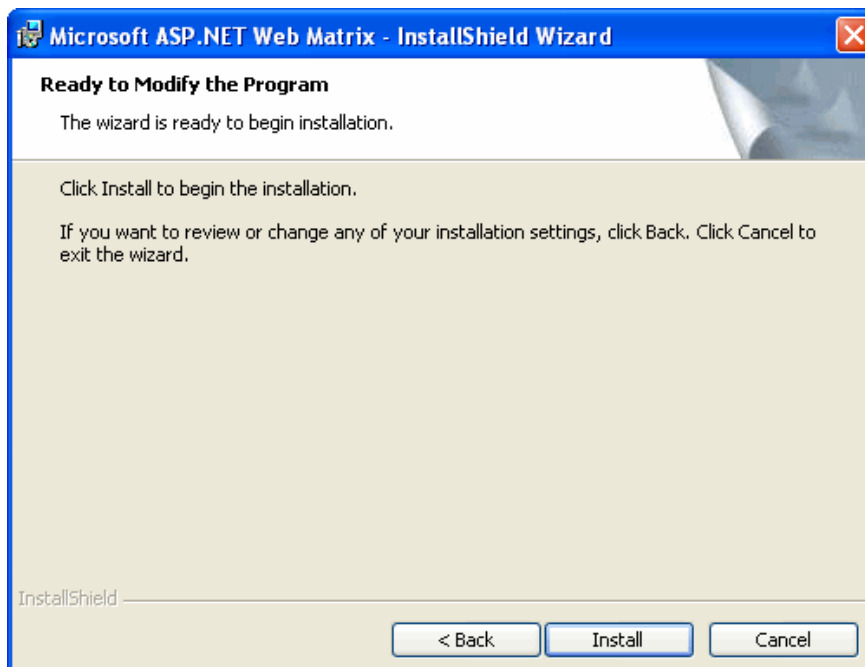
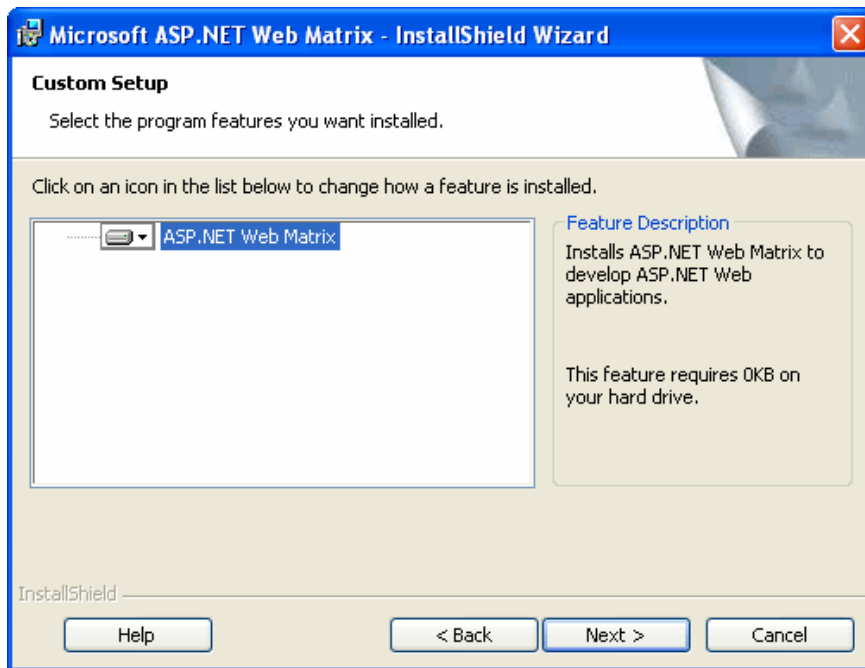
Sau khi tải xuống, ta nhấp đôi vào (double-click) **webmatrix.msi** (được chuẩn bị cài với MS Installer), Web Matrix sẽ được mở ra và cài đặt như sau:

1. MS Installer sẽ dùng InstallShield Wizard để khởi động việc cài Web Matrix:



2. Nhấp **Next** và chọn **Install**. Nếu ta đã cài Web Matrix, chọn Modify để cài lại hoặc chọn Repair để sửa chữa Web Matrix. Sau đó theo hướng dẫn để chọn chỗ chứa ứng dụng và hoàn tất việc cài đặt.



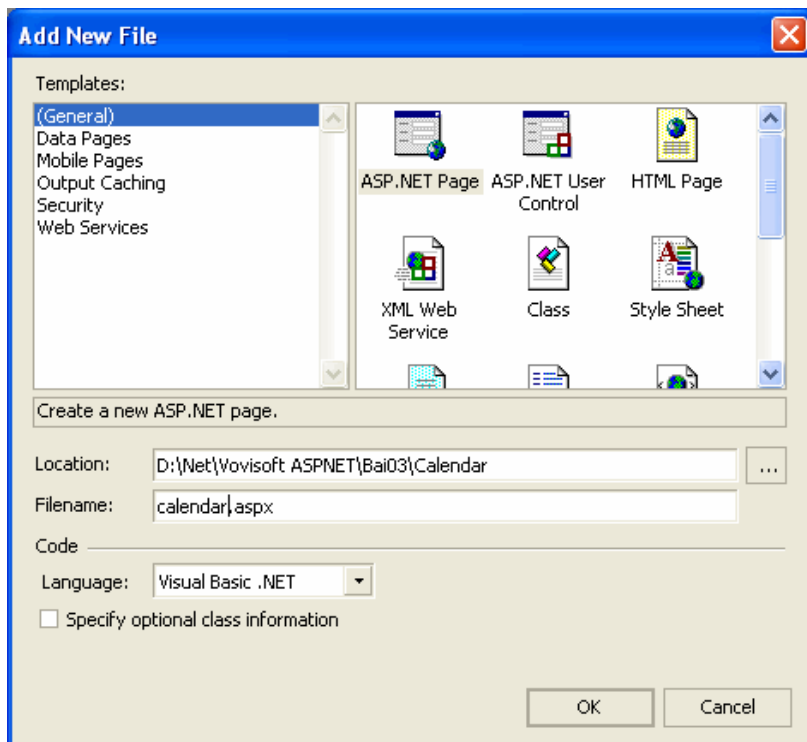


ĐI DẠO 1 VÒNG VỚI WEB MATRIX

1. Ta khởi động Web Matrix bằng cách nhấp đôi **ASP.NET Web Matrix icon** trên Desktop:



2. Web Matrix sẽ mở ra IDE với **Add New File** windows như sau:

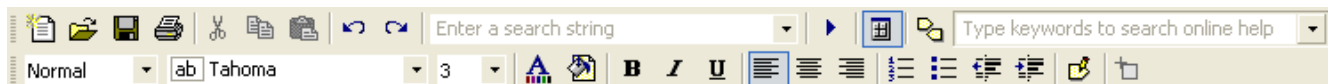


3. Ta có thể tạo 1 trang ASP.NET với Filename **calendar.aspx** ở Location **D:\Net\Vovisoft ASPNET\Bai03\Calendar** để có thể đi dạo 1 vòng với Web Matrix.

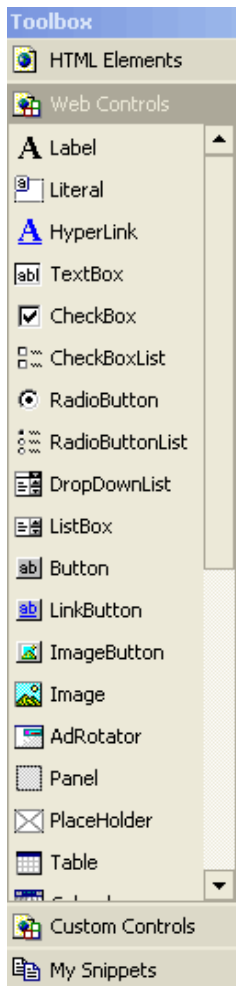
4. Trước hết, ta nhận thấy Menu Bar nằm phía dưới nhãn hiệu ứng dụng (Title Bar) Web Matrix rất quen thuộc vì theo tiêu chuẩn chung của các sản phẩm MS Windows. Menu Bar này gồm nhiều bộ thực đơn, dưới có các thanh thực đơn (Menu Items) thích hợp cho từng công dụng khác nhau và các thanh này cũng xuất hiện hay dấu biến đi tùy theo chức năng của các trang Web hay dạng (Mode) được dùng trong môi trường IDE.



5. Dưới đó, ta thấy Toolbar tiêu chuẩn của Web Matrix. Standard Toolbar này gồm 1 bảng chứa mọi công cụ dùng cho việc phát triển các ứng dụng ASP.NET



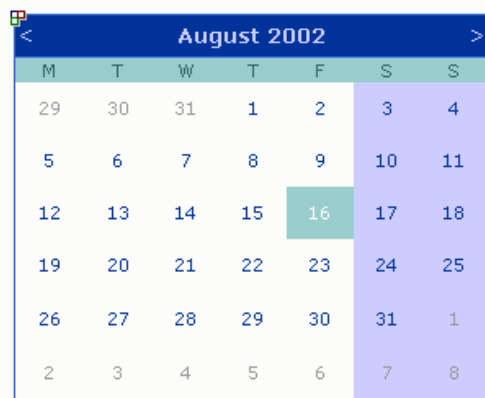
6. Hộp công cụ (Toolbox) nằm phía trái của Windows chính, cho phép ta chọn và kéo lê (drag) các controls vào trong trang Web, Form hay chứa các công cụ do ta đặc chế hoặc do nhập từ bên ngoài. Toolbox gồm có 4 ngăn: HTML Elements (cho các HTML tags), Web Controls, Custom Controls và My Snippet.



7. Windows chính của Web Matrix gọi là **Document Window**, nơi đây hiển thị trang Web ta phác hoạ. Window này tự động tạo ra khi ta mở hay tạo 1 ứng dụng của Web Matrix.

Phía bên dưới Document Window gồm 4 ngăn (tabs) cho 4 dạng (mode) riêng biệt của cùng 1 trang Web: Design, HTML, Code and ALL (kết hợp giữa dạng HTML và Code).

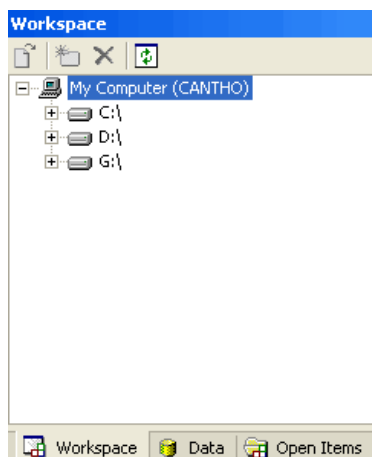
Create Calendar with Web Matrix



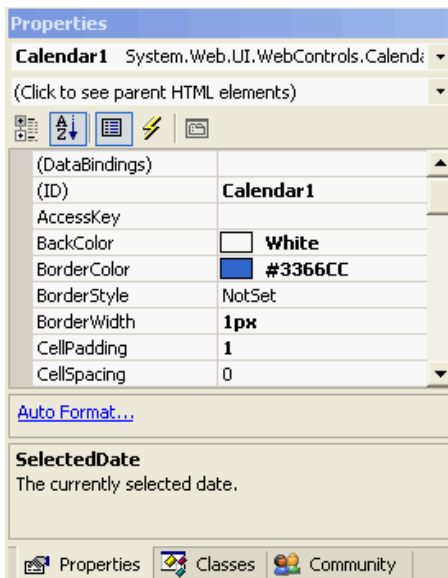
August 2002						
M	T	W	T	F	S	S
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

8. Workspace Window nằm phía trên bên phải gồm có 3 ngăn: Workspace, Data và Open Items.

- Workspace Window là nơi ta tổ chức hay quan sát File System (Hệ Thống Tập Tin), các Directories, các tập tin ở đĩa cứng cục bộ (local hard drives) hay mạng (network drives), ...
- Data Window dùng để nối các cơ sở dữ liệu (Database), mở các tables hay lưu trữ các stored procedures. Nếu ta không cài MS SQL Server 2000, Web Matrix cung cấp nối để tải và cài MSDE (MS Data Engine) dùng với MS Access thay vì SQL 2000.



9. Properties Window nằm phía dưới bên phải gồm có 3 ngăn: Properties, Classes và Community. Properties Window dùng để định rõ tính chất và cách biểu hiện trang Web, document và các controls của Web Matrix. Nơi đây cũng là nơi ta quan sát và truy tìm các .NET Assemblies hay 'nổi ... vòng tay lớn' tới cộng đồng ASP.NET để tìm kiếm các trợ giúp cần thiết.



TẠO TRANG ASP.NET VỚI WEB MATRIX

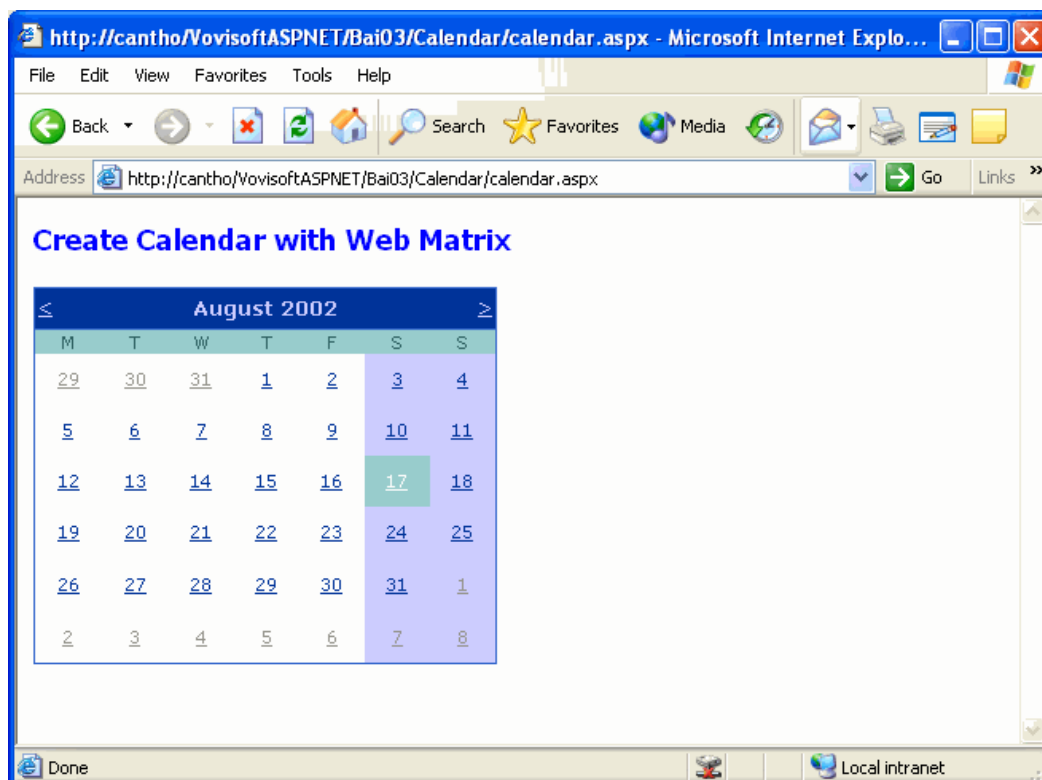
Việc tìm hiểu Web Matrix là việc làm lâu dài, ta sẽ tham khảo thêm khi học đến các phần mục khác của ASP.NET tý như về ASP.NET Objects, Web Form, Database, XML, Web Services, ...

Tới đây, ta thử tạo vài trang ASP.NET với Web Matrix. Mong rằng, trong khi đi dạo với Web Matrix, ta đã thu lượm một vài điều hữu ích cho các bài tập sau đây:

Bài tập 1:

Mục đích:

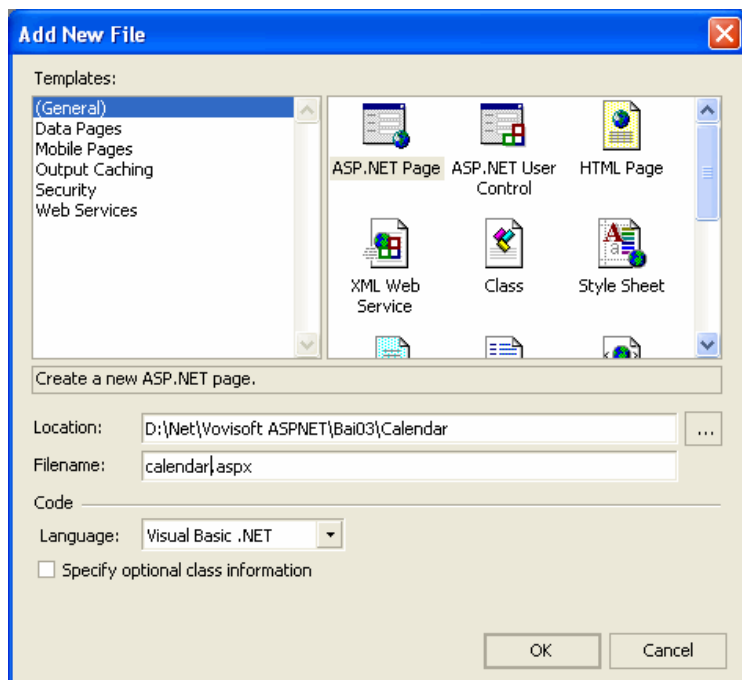
Trong bài tập này, ta sẽ xây dựng một trang ASP.NET với Web Matrix dùng hiển thị 1 cuốn lịch chỉ rõ ngày, tháng, năm như sau:



Các bước thứ tự như sau:

1. Chạy ứng dụng Web Matrix. Trong Add New File window, chọn General ở Templates pane và chọn mẫu ASP.NET Page. Gõ vào hộp chữ Location hay dùng nút Browse để chọn **D:\Net\Vovisoft ASPNET\Bai03\Calendar** và tên của trang Web là **calendar.aspx** và nhấp nút OK.

Templates (General)	ASP.NET Page
Location	D:\Net\Vovisoft ASPNET\Bai03\Calendar
Filename	calendar.aspx
Language	Visual Basic.NET

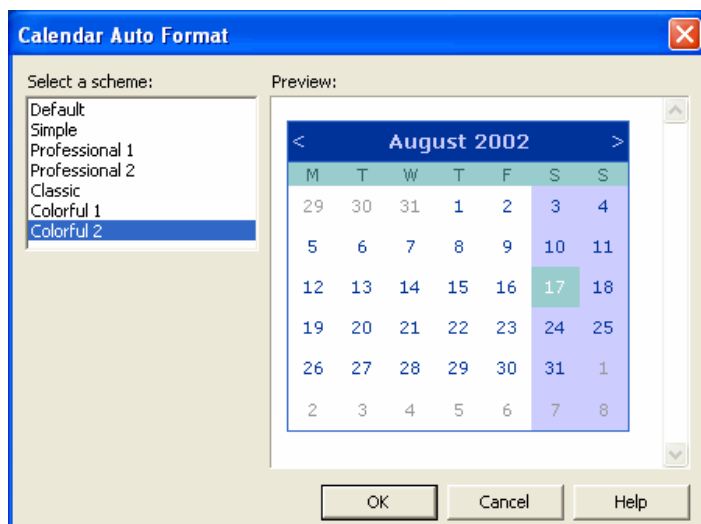


2. Trang Web calendar.aspx sẽ mở ra dưới dạng Design và sẵn sàng cho ta xử dụng. Từ hộp công cụ (Toolbox), ta kéo lê (drag) một công cụ Nhãn Hiệu (Label) vào Document Window và nhấn nút Enter để mở đầu 1 đoạn văn (paragraph) mới. Sau đó, ta kéo lê công cụ Calendar vào và đặt dưới nhãn hiệu trên.

Ta định tính chất của công cụ nhãn hiệu đó như sau:

Label ID	lblTitle
Fonts	Medium
ForeColor	Blue
Text	Create Calendar with Web Matrix

3. Chọn công cụ Calendar trong Document window. Nhấp **Auto Format ...** (nằm bên dưới Properties Window) và chọn kiểu mẫu **Colorful2** như sau:

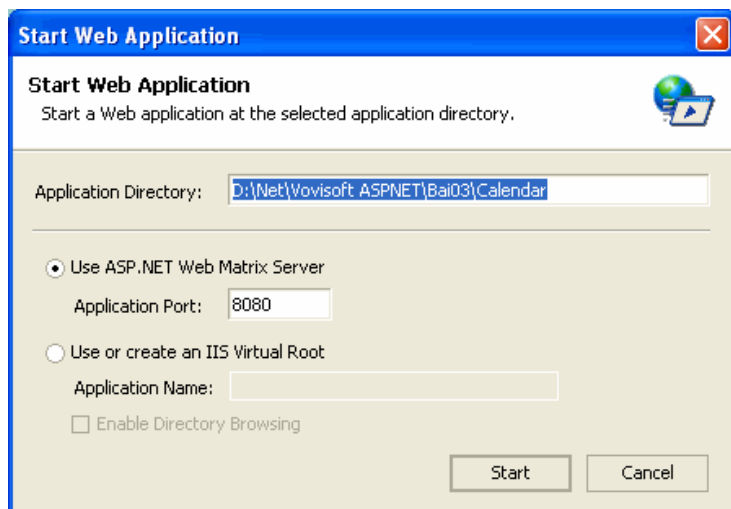


4. Nhấp nút **Run** để chạy ứng dụng. Nút **Run** này nằm trên Toolbar, ở giữa 1 hộp chữ và nút **Toggle Toolbox**:

Enter a search string

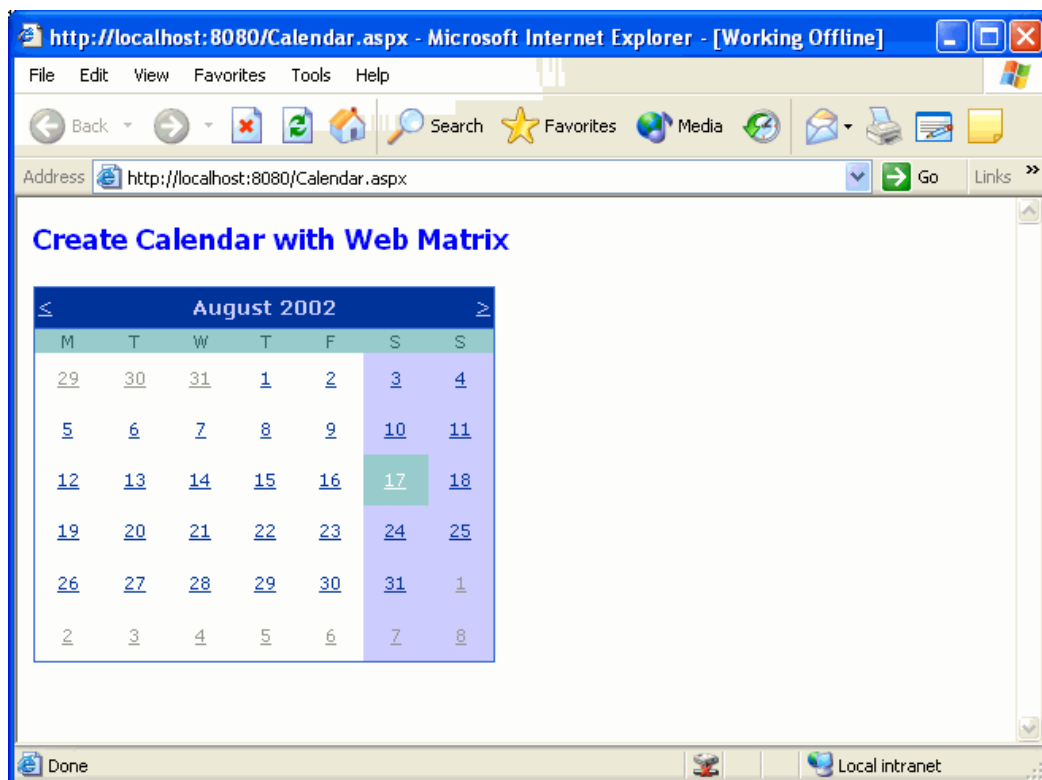
5. Ta có 2 sự lựa chọn để chạy trang Web:

- Cách 1: Dùng **Web Matrix Server** chạy trang Web trong môi trường của Web Matrix với **port** mặc định là 8080.
- Cách 2: Dùng **IIS (Internet Information Server)** để tạo Application (ta có thể dùng virtual directory **VovisoftASPNET** mà ta bố trí trước đây) để chạy trang Web trong MS Internet Explorer (IE).

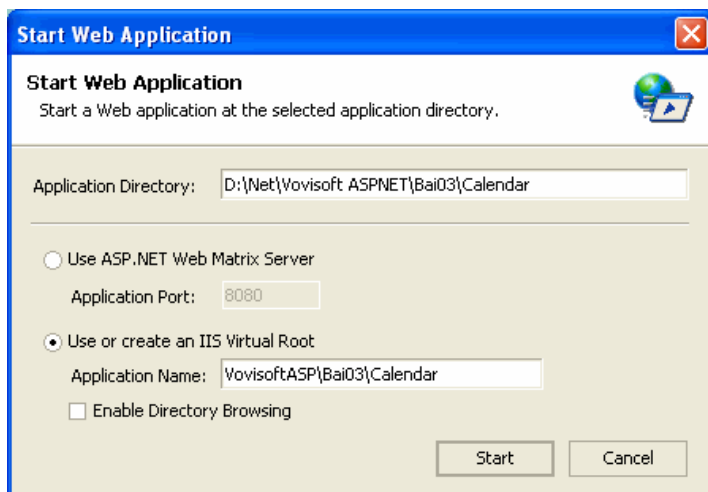


Khi dùng cách 1 để chạy đi chạy lại trang Web để điều chỉnh hay sửa lỗi, để ý ta phải ngừng (stop) Web Matrix Server để chạy lại, nếu không ta sẽ tạo ra quá nhiều instances của Web Matrix Server có thể làm rỉ memory (memory leaking).

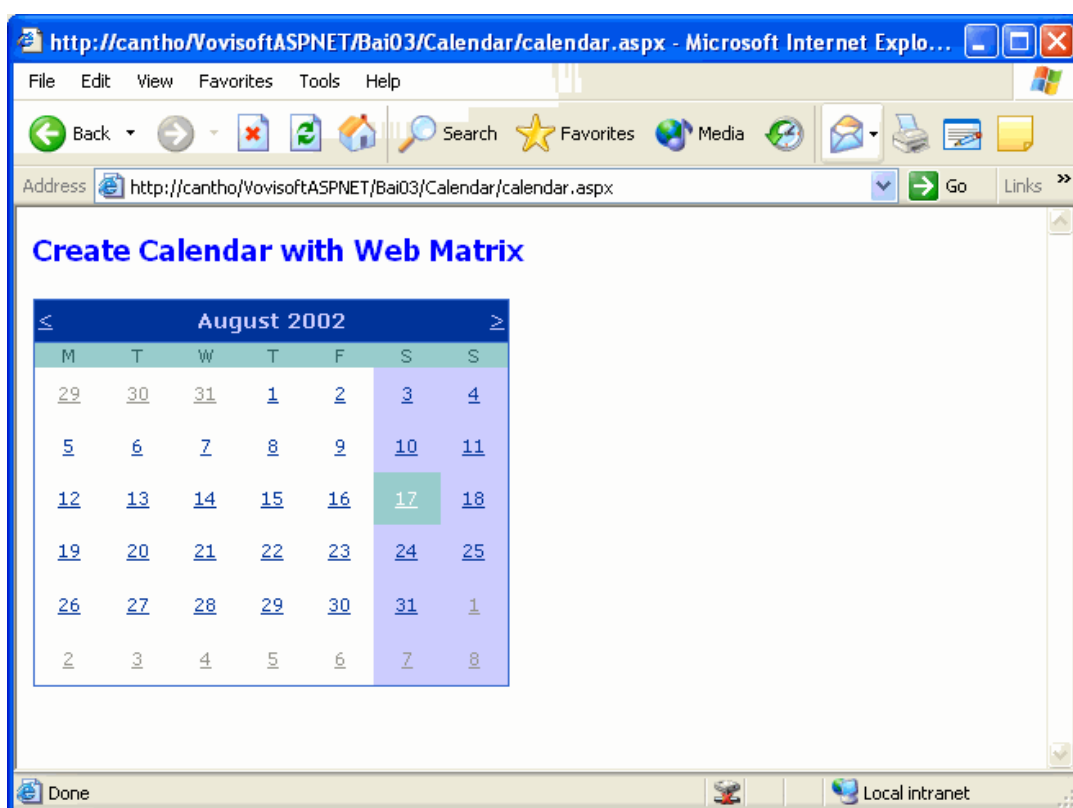
Đây là kết quả hiển thị trang Web cách thứ nhất với localhost và port 8080 được viết dưới dạng **localhost:8080**



Còn cách thứ nhì được dùng hay tạo virtual directory trong IIS như sau:



và hiển thị dùng IE như sau (trong trường hợp này, ta phải gõ đúng địa chỉ (URL) của trang Web calendar.aspx mới có thể hiển thị như ý):



Tóm tắt

Như ta đã có dịp làm quen và so sánh giữa các phương pháp xử dụng các công cụ cho việc phát triển các trang Web khác nhau, từ Notepad, Visual Studio.NET và bây giờ Web Matrix. Có nhiều người ngại dùng Web Matrix vì tính cách cộng đồng của nó và nhất là không được yểm trợ trực tiếp của Microsoft. Nhờ là Web Matrix được phác thảo (design) để chuyên trị các trang ASP.NET, còn nếu muốn dùng cho các ứng dụng khác, đâu ai ngăn cấm ta dùng Visual Studio.NET. Trong môi trường phát triển ứng dụng của các cơ sở chính phủ, các công ty thương mại hay các ngân hàng, ta nên dùng Visual Studio.NET thay vì Web Matrix vì được yểm trợ của Microsoft nhưng để tự học (trong sở hay tại nhà) hay chuẩn bị mẫu các trang ASP.NET, Web Matrix là nhận tuyển thích hợp nhất, càng xử dụng Web Matrix càng lâu, tình nghĩa càng thêm ... đậm đà thâm thía. Riêng tôi thì tôi đã ... 'ghiền' Web Matrix rồi nên 'còn ... chút gì để nhớ, để quên' nữa mà nói, mà bàn.

Download Source Code

References

- Tải [Web Matrix](#) ở đây.
 - Bạn có tham khảo thêm về Web Matrix cũng như các Tutorials dùng Web Matrix được soạn thảo do [nhóm chủ trương và thiết kế Web Matrix](#). Ở đây, cũng có Diễn Đàn giải đáp các thắc mắc hay trở ngại của bạn khi dùng Web Matrix tỷ như việc cài đặt Web Matrix, việc cài đặt MSDE, ...
-

Bài làm ở nhà

Câu hỏi 1: Web Matrix có gì ưu việt so với Notepad, Visual Studio.NET?

Câu hỏi 2: Kể vài đặc trưng của Web Matrix?

Bài làm 1: Dùng Web Matrix để phát triển 1 trang ASP.NET trong đó bạn tập khai phá và xử dụng tất cả các công cụ có sẵn trong hộp công cụ (Toolbox) từ HTML, Server Controls cũng như làm quen cách xác định tính chất đặc trưng cho từng control một với Properties Windows.

Bài làm 2: Tải MSDE (sau khi cài Web Matrix, vào trong Workspace Windows nhấn Database để truy tìm nổi tải MSDE), cài đặt MSDE vào máy vì tính của bạn cũng như tham khảo các giải đáp liên hệ đến sự trở ngại khi cài đặt MSDE.

Bài 04

Dùng ASP.NET Objects với VB.NET (Part I)

Đố ai biết lúa mấy cây
Biết sông mấy khúc, biết mây mấy tầng
Đố ai quét sạch lá rừng
Đề ta khuyển gió, gió đừng rung cây
Rung cây, rung cỏ, rung cành
Rung sao cho chuyển lòng anh thương nàng
Ca Dao Việt Nam

Đố ai có thể tham khảo các đối tượng dùng trong ASP.NET mà không có khái niệm về phương pháp lập trình theo khuynh hướng đối tượng (**Object Oriented Programming**). Các bài viết về [OOP và Visual Basic.NET](#) do **thầy Lê Đức Hồng** soạn được trình bày ở trang nhà, ta nên tìm hiểu trước để làm hành trang cho việc dùng ASP.NET Objects.

Thật ra, nếu ASP.NET chỉ dùng ngôn ngữ lập trình VB.NET cùng với những đặc trưng của ngôn ngữ này thay vì VBScript hay JavaScript để phát triển (develop) trang Web thì ta sẽ không lấy gì hứng thú cho lắm. Điều quan trọng ở đây là .NET Framework bao gồm cả ASP.NET trong cấu trúc nền, do đó ASP.NET khai thác được mọi tài nguyên mà .NET Framework cung ứng gồm cả hàng trăm classes (built-in classes) cũng như hàng nghìn đối tượng (object) xây dựng sẵn, giúp ta nâng cao (enhance) và mở rộng các chức năng (functionality) các trang Web 1 cách dễ dàng hơn.

Trong bài 'Dùng ASP.NET đối tượng (object) với VB.NET', ta sẽ lần lượt tìm hiểu:

- Đối tượng (object) cơ bản và đặc tính (properties)
- Đối tượng ASP.NET Objects và phương pháp khai thác các đối tượng
- Phương pháp làm việc với Session và Cookies

OBJECTS CƠ BẢN

Đối tượng (Object)

Như ta đã biết (hay biết nhưng đã quên ?), Objects biểu hiện cho một cụm (hay nhóm, bộ phận) nguồn mã có thể tái sử dụng (reusable code) trong đó định nghĩa rõ ràng và đầy đủ các loại (**classes** - lớp hay hạng) đối tượng (object) là gì cũng như phương pháp để dùng các đối tượng (object) cùng với các dữ liệu (data) của nó.

Ở .NET Framework, ta có thể tìm thấy đủ loại (classes) định nghĩa rõ ràng các đối tượng (object) dùng cho ASP.NET nhưng trong thực tế, có nhiều loại (classes) không dùng hay liên quan gì đến ASP.NET mà nếu muốn, ta vẫn có thể dùng được như thường vì như đã trình bày, ASP.NET là 1 bộ phận trong cấu trúc của .NET Framework .

Đặc tính (Properties)

Properties là các biến số dùng miêu tả đối tượng (object), tỷ như cây kim giờ, kim phút và kim giây dùng để miêu tả cái đồng hồ chẳng hạn.

Methods

Methods là các phương pháp để dùng đối tượng (object), tỷ như bố trí giờ, lên giấy thiều hay thay pin cho đồng hồ nêu ở trên.

Object Instances

Trước khi ta có thể sử dụng một đối tượng (object), ta cần tạo ra một instance giống như ta đúc khuôn (class, object) để tạo ra một bức tượng cá biệt nào đó (instance). Như vậy, khi lập trình, ta nhớ bố trí một biến số (variable) để nắm giữ một instance trong bộ nhớ (memory) của máy vi tính. Lưu ý là ta phải phân biệt được sự khác nhau giữa 1 đối tượng thật sự (actual object) và 1 đối tượng định hình (instance), bằng không chắc chắn ta sẽ vào 'mê hồn trận' không lối thoát.

Trở lại thí dụ về cái đồng hồ, đồng hồ nói chung là ... 'cái đồng hồ', hay gọi là 'generic object'. Đồng hồ **của ta** là 1 instance của cái đồng hồ (dĩ nhiên ta sẽ không cho phép ai đó ... cầm nhầm). Đồng hồ của bạn ta cũng là 1 instance. Cả hai tuy đều là đồng hồ nhưng có đặc tính khác nhau, tỷ như đồng hồ ta bằng vàng, đồng hồ bạn ta bằng bạc, ...

Nếu dùng VB.NET:

```
Clock objClock as New Clock()
```

trong đó:

- **Clock** là đối tượng (object)
- **objClock** là biến số (variable) dùng nắm giữ đối tượng định hình (instance) tạo ra từ mệnh lệnh New Clock()

Static Member

Thành viên cố định (static member) được dùng trực tiếp để bố trí đặc tính (properties) hay gọi methods trong 1 object. Trở lại thí dụ về cái đồng hồ, tỷ như ta bố trí 1 máy đếm (counter) trong khuôn (class) đồng hồ để kiểm tra xem người dùng đã đúc bao nhiêu cái đồng hồ (clock instances) chẳng hạn. Điều đáng để ý là counter đó không thuộc về 1 instance nào hết - nó được gắn dính liền trong cái khuôn đồng hồ. Để công bố (declare) đặc tính trực tiếp, ta dùng keyword **static**.

ASP.NET OBJECTS

Trong hàng nghìn ASP.NET Objects, ta chỉ có thể xem xét 1 số nhỏ tiêu biểu như sau:

- Response Object
- Request Object
- Page Object
- Session Object
- HttpCookie Object
- HttpApplication
- HttpServerUtility Object

Response Object

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

Phương pháp (Method) Write

Thí dụ sau đây dùng phương pháp (method) **Write** của Response object để hiển thị vài hàng chữ ở Client browser:

```
<%@ Page Language="VB" %>

<script runat="server">
sub Page_Load(obj as object, e as eventargs)
dim i as integer

'Dùng phương pháp (method) Write để hiển thị vài hàng chữ sau đây
Response.Write("Using Write method of Response object")
Response.Write("<HR width=100%>")
Response.Write("<font size=" & 3 & ">Hello Vovisoft<br></font>")

end sub
</script>

<html>
<body>

</body>
</html>
```

Chú thích:

Mỗi khi user yêu cầu hiển thị 1 trang Web, ASP.NET tạo ra 1 đối tượng định hình (instance) từ đối tượng (Object) HttpResponse có chứa đầy đủ đặc tính và phương pháp (method) cần thiết để thông tin với Client browser. Tên cố định của instance đó là **Response** trong đó có phương pháp (method) Write dùng để viết và hiển thị 1 hàng chữ ở browser.

Đề ý ký hiệu \> ở cuối mã sau đây. Nếu ta không dùng ký hiệu \ để phân biệt 100%\> và 100%>, ASP.NET sẽ tưởng ta dùng %> để chấm dứt phần script block và sẽ tạo lỗi.

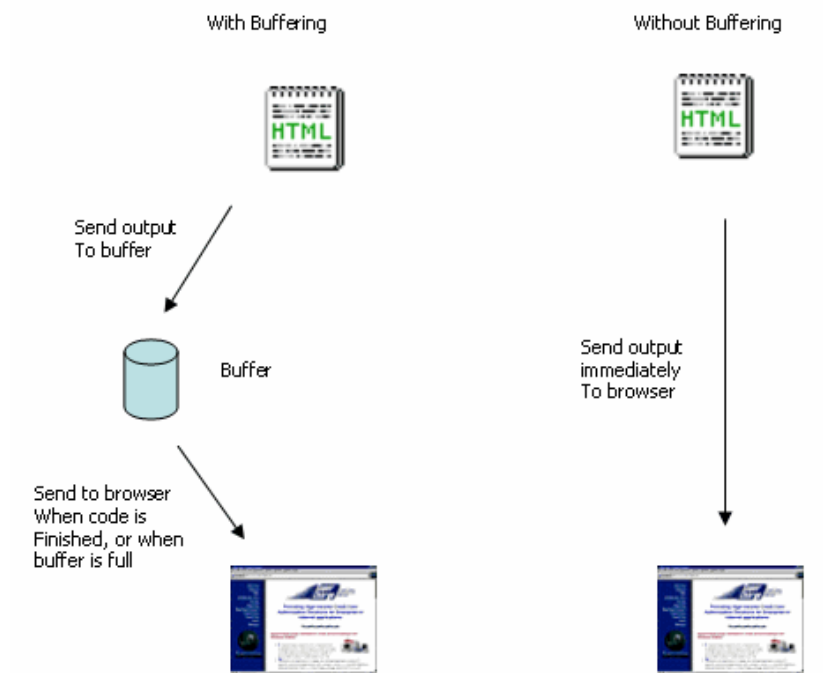
```
Response.Write("<HR width=100%\>")
```

Ký hiệu \ là 1 character sequence đặc biệt với tên thông dụng **escape character** dùng để phân biệt các ký hiệu đặc biệt có thể trùng với các ký hiệu của ASP.NET hay của VB.NET nhưng nếu muốn hiển thị double quotes, ta dùng double quotes double quotes.

```
Response.Write("<HR width=" "100%" ">")
```

Buffering Pages

ASP.NET dùng **buffer** để kiểm soát (control) khi nào gửi sản phẩm (output) tới browser. Đây cũng là kiểu mẫu chính (default method) ASP.NET dùng, khi output được buffered, ASP.NET chờ cho tới khi nào tất cả nguồn mã được thi hành mới gửi tới browser ... 'một lần rồi thôi'. Trong kiểu **unbuffered output**, mã sẽ gửi tới browser từng cụm hay từng phần một - ở đây muốn nói tới 'output of each method'.



Nếu ta muốn tắt buffer (turn buffering off), bố trí mã Buffer = false **trước khi** gọi đi browser:

```
<% Response.Buffer = False %>

<html>
<body>

...

...

</body>
</html>
```

vì **không thể** vừa tắt buffer (trong mã HTML) vừa gọi đi theo mã này được:

```
<html>
<body>

<% Response.Buffer = False %>

</body>
</html>
```

Để vận dụng hay xử dụng buffer, ta dùng phương pháp (method):

- Clear
- Flush
- End

Phương pháp (method) **Clear** dùng để xóa (clear) buffer, **Flush** dùng để gửi tức khắc nội dung buffer, còn **End** dùng để chặn (stop) không cho buffer gửi sản phẩm mới đi mà chỉ cho phép gửi những gì đang chứa trong buffer mà thôi như thí dụ sau:

```
<%@ Page Language="VB" %>
```

```

<script runat="server">

sub Page_Load(obj as object, e as eventargs)
    dim i as integer

    Response.Write("Before flush<br>")
    Response.Flush()
    for i = 0 to 5000
        'just wasting time
    next

    Response.Write("After flush, before clear<br>")
    Response.Clear()
    for i = 0 to 5000
        'just wasting time
    next

    Response.Write("After clear, before end<br>")
    Response.End
    for i = 0 to 5000
        'just wasting time
    next

    Response.Write("After end<br>")
end sub

</script>

<html>
<body>

</body>
</html>

```

Redirecting Users

Phương pháp (method) Redirect dùng để chuyển user qua trang Web khác 1 cách gián tiếp, tỷ như:

```

<%@ Page Language="VB" %>

<script language="VB" runat="server">

sub Page_Load(obj as object, e as eventargs)
    Response.Redirect("http://www.vovisoft.com")
end sub

</script>

<html>
<body>

Hello

</body>

```


</html>

Trang này sẽ chuyển user tới trang nhà của Vovisoft trước khi user thấy được chữ 'Hello'. Đây cũng là cách thức ta dùng để chuyển user qua 1 trang Web khác sau khi xác định tên và mật mã (username and password) của user trong 1 form đăng hạn.

Request Object

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

Tuy vậy, object này không cần thiết vì ASP.NET đã lo lắng, đảm đương hầu hết trách nhiệm thông tin giữa Server và Client browser dùm ta. Nếu cần thì ta cứ việc dùng, không sao cả, càng vui (the object is still there for your use).

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

Ta đơn cử 1 thí dụ dùng **querystring** như sau:

`http://www.vovisoft.com?username=NangVu&password=forget`

trong đó, username=NangVu và password=forget là 2 dữ kiện (data) dưới dạng cặp key/value (**key/value pairs**). **username** và **NangVu** là khoá và giá trị thứ nhất, **password** và **forget** là khoá và giá trị thứ nhì, ... Còn **?** chỉ thị cho biết có thông tin đính kèm và **&** dùng phân biệt các cặp key/value với nhau. Như vậy, ta biết có tổng cộng 2 dữ kiện (data) được chuyển đi từ Client browser.

Lưu ý:

Thường thì Querystring **chỉ dùng tối đa 255 characters**.

Server có thể tìm lại dữ kiện từ querystring đó như sau:

```
'Tóm lại toàn bộ thặng tin "username=NangVu&password=forget"
Request.QueryString
```

```
'Tìm lại giá trị (value) "NangVu"
Request.QueryString("username")
```

Nếu user đệ trình dưới dạng **form**:

```
'Tóm lại toàn bộ giỏ trị trong form
Request.Form
```

```
'Tìm lại giá trị value "NangVu"
Request.Form(username)
```

Tóm lại, cả 2 kiểu đều được dùng để thu thập thông tin từ Client browser. Nhưng, ASP.NET dùng **Web Form Framework** để chăm nom, sắp đặt mọi yêu cầu (request) của Client browser cho ta, nhờ vậy ta có thể ... 'quảng gánh lo đi' để lo chuyện ... 'bao đồng', ủa quên ... chuyện quan trọng khác. Ta sẽ tham khảo về **Web Form** ở các bài sau.

Cũng cần nhắc ở đây, ta có thể dùng Request để thu thập thông tin về **ServerVariables** và **Cookies**. **ServerVariables** chứa thông tin của Server tỷ như **IP address** hay **HTTP protocol**, ... **Cookies** (dưới hình thức 1 tập tin) chứa thông tin cần thiết ở chính máy vi tính của user (Client's computer).

Dưới đây vài biến số (variables) thông dụng trong bộ ServerVariables Collection:

Variable	Cung dụng
URL	chứa thông tin về URL của trang Web
PATH_INFO	tương tự như URL
PATH_TRANSLATED	physical path của ASP.NET ở Server site
SERVER_NAME	tên Web Server, tỷ như cantho
SERVER_SOFTWARE	tên và phiên bản của Web Server, tỷ như Microsoft-IIS/5.0

Page Object

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

Trong khái niệm về **OOP (Object Oriented Programming)**, 1 khuôn mẫu (class) định nghĩa mọi đặc tính và phương pháp (method) cần thiết cho class đó cũng như **kế thừa** mọi đặc tính và phương pháp (method) của cha mẹ hay tổ tiên hoặc có thể tạo ra thành viên **con, cháu**, ... (inherit its member and also create your own members for the child class).

Áp dụng lối suy nghĩ như vậy vào các trang ASP.NET. Ta thấy, các trang ASP.NET là child objects của **Page** object. Mọi đặc tính (properties) và phương pháp (method) ta định nghĩa trong trang ASP.NET trở thành thành viên trang Web, như vậy khi ta tạo ra 1 trang Web khác từ trang Web thứ nhất, nó sẽ thừa kế mọi đặc tính (properties) và phương pháp (method) theo kiểu ... 'cha truyền con nối'.

Page object gồm vài thành viên cơ bản, tỷ như:

- Load
- IsPostBack
- DataBind

Load dùng để khởi động (fire, start or activate) khi trang Web bắt đầu hiển thị ở browser. **IsPostBack** cho ta biết form ở trang Web đã được gửi đi tới cùng trang Web hay không? Còn **DataBind** nối kết mọi dữ kiện (data) từ cơ sở dữ liệu (database) với công cụ (controls) ở trang Web. Ta sẽ tham khảo DataBind ở bài nói về Database.

Bài tập 1:

Mục đích:

Trong bài tập này, ta tập cách sử dụng **Load event** ở trang ASP.NET. Thật ra, ta có thể thi hành được nhiều việc với sự cố này tỷ như xác minh lý lịch user, chuyển data từ cơ sở dữ liệu (database) hay chuyển user qua 1 trang Web khác, ... Ở đây, ta muốn hiển thị câu chào xã giao tùy theo giờ giấc mỗi khi user lướt trang Web của ta như sau:

Các bước thứ tự như sau:

1. Chạy ứng dụng Notepad và gõ nguồn mã sau:

```
<%@Page Language="VB" %>
```

```
<script runat="server">
```

```

sub Page_Load(obj as object,e as EventArgs)
    dim Now as DateTime = DateTime.Now
    dim intHour as integer = Now.Hour

    lblCauChao.Text ="Bây giờ là " & _
    Now.ToString("T")+ "<p>"

    if intHour <12 then
        lblCauChao.Text += "Chào bạn. Mới sáng sớm mà đã lướt mạng rồi sao? Cảm ơn nhiều."
    elseif intHour >12 &intHour <18 then
        lblCauChao.Text += "Buổi trưa vừa ăn vừa lướt e đau bao tử đấy."
    else
        lblCauChao.Text += "Buổi tối thường cho giờ tâm tình. Bạn đã sẵn sàng chưa?"
    end if
end sub

</script>

<html>
<body>

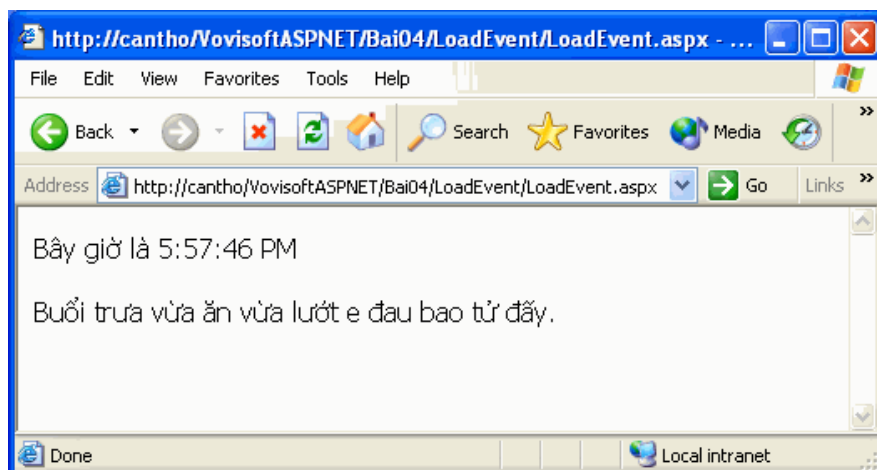
<form runat="server">
<asp:Label id="lblCauChao" runat="server"/>
</form>

</body>
</html>

```

2. Lưu trữ tập tin với tên LoadEvent.aspx ở folder 'D:\Net\Vovisoft ASPNET\Bai04\LoadEvent' và chạy IE với URL như sau: <http://cantho/VovisoftASPNET/Bai04/LoadEvent/LoadEvent.aspx>

Nhớ khi lưu trữ, save dưới dạng UTF-8 để có thể hiển thị tiếng Việt như sau:



3. Kiểm tra câu chào có thích hợp hay không bằng cách thay đổi giờ của máy vi tính trước 12 giờ trưa, từ 12 giờ trưa đến 6 giờ chiều và sau 6 giờ chiều khi chạy trang Web.

Phần Chú Thích:

```

dim Now as DateTime = DateTime.Now
dim intHour as integer = Now.Hour

```

Trong sự cố Load ở subroutine Page_Load, ta bố trí biến số (variable) Now và intHour để ấn định thời gian hiện tại và giờ với đối tượng (Object) DateTime. Để ý, Now (cho biến số) trùng với đặc tính (properties) Now của đối tượng (Object) DateTime nhưng chức năng của chúng khác biệt nhau. Đây cũng là 1 kinh nghiệm về lập trình ta cần lưu ý. **Nhớ ngôn ngữ lập trình VB.NET không theo quy ước 'case sensitive',** như vậy đối tượng (objects) mang tên MyProducts hay myproducts cũng chỉ là một mà thôi.

```
Now.ToString("T")+"<p>"
```

ToString("T") là phương pháp (method) chuyển dạng 'Thì Giờ' thành hàng chữ (string) để hiển thị. Ở đây, ta nhận thấy có 1 argument "T" biểu hiện cho 'giờ' dưới dạng **3:07:30 PM** chẳng hạn.

Dưới đây là bảng liệt kê các **arguments** dùng cho đối tượng (objects) **DateTime**:

String as argument	Example
"d"	1/26/2002
"D"	Thursday, January 26, 2002
"f"	Thursday, January 26, 2002 5:30 PM
"F"	Thursday, January 26, 2002 5:30:45 PM
"g"	1/26/2002 5:30 PM
"G"	1/26/2002 5:30:45 PM
"m"	January 26
"r"	Thu, 26 January 2002 17:30:45 GMT
"s"	2002-01-26T17:30:45
"t"	5:30 PM
"T"	5:30:45 PM
"u"	2002-01-26 17:30:45Z
"U"	Thursday, January 26, 2002 17:30:45
"y"	Januray, 2002
"dddd, MMMM dd yyyy"	Thursday, January 26 2002
"ddd, MMM d ""yy"	Thu, January 26' 02
"dddd, MMMM dd"	Thursday, January 26
"M/yy"	1/02
"dd-MM-yy"	26-01-02

Session Object

Trong bài 01 'Làm Quen với ASP.NET', ta đã biết một khi Client đã nhận được thông tin (information) từ Server rồi, quá trình trao đổi qua lại đó kết thúc ngay tức khắc. Sau đó, Server và Client trở thành ... 'người xa lạ', coi như là chưa từng bao giờ gặp nhau (stateless model), ta gọi là kiểu ... 'làm ngơ' và như vậy, khi gặp lại, ta biết là có ... 'duyên nợ' gì với nhau hay không là nhờ ở đối tượng (objects) **Session**.

Session object cho phép ta lưu giữ thông tin về ... 'người ấy' dưới các dạng như biến số (variables), objects, strings hay ... bất cứ loại thông tin nào có dính dáng tới họ tại một chỗ nào đó ở Server trong lúc họ 'rong chơi' trong site của ta. Chỉ khi nào người lướt mạng bỏ ta 'sang ... ngang' hay nhảy qua 1 site khác, bấy giờ Session mới kết thúc, mọi thông tin về ... 'người ấy' đều được xoá sạch và người ấy trở thành ... 'cô nhân', hay đúng hơn, là ... 'người dưng'.

Bài tập 2:

Mục đích:

Dùng Session object để lưu trữ thông tin người lướt mạng và dùng thông tin đó để đối thoại hỗ tương với nhau.

Các bước thứ tự như sau:

1. Chạy ứng dụng Notepad và gõ nguồn mã sau:

```
<%@Page Language="VB" %>

<script runat="server">

sub Submit_Click(obj as object,e as EventArgs)
    if tbName.Value <>""
        Session("Name ")=tbName.Value
        Response.Write("Hi " & Session("Name ")&"!")
    else
        Response.Write("You forgot to enter a name.")
    end if
end sub

</script>

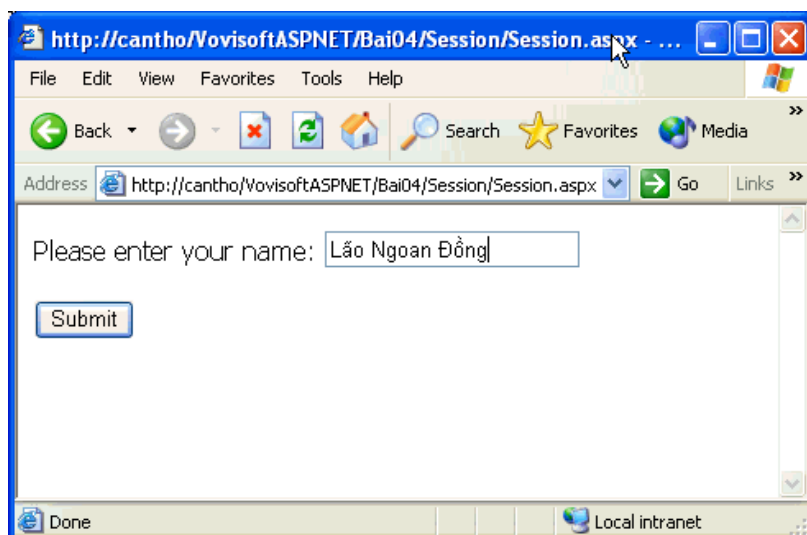
<html>
<body>

<form runat="server">
Please enter your name:
<input type="text" id="tbName" runat="server"/>

<p>
<asp:Button id="btSubmit" text="Submit" runat="server" OnClick="Submit_Click" />
</form>

</body>
</html>
```

2. Lưu trữ tập tin với tên Session.aspx ở folder 'D:\Net\Vovisoft ASPNET\Bai04\Session' và hiển thị trang Web để kiểm tra phương pháp (method) làm việc của đối tượng (objects) Session với URL sau: **<http://cantho/VovisoftASPNET/Bai04/Session/Session.aspx>**



Phần Chú Thích:

```
if tbName.Value <>""
    Session("Name ")=tbName.Value
    Response.Write("Hi " & Session("Name ")&"!")
else
    Response.Write("You forgot to enter a name.")
end if
```

Ở phần Script, ta kiểm tra 1 cách đơn giản xem user có thật sự gõ chữ nào vào trong hộp chữ 'tbName' hay không? Ở đây, ta tạm bỏ qua việc **validation** (phê chuẩn hiệu lực) và sẽ tìm hiểu thêm ở các bài sau. Khi user gõ tên vào hộp chữ, ta tạo 1 biến số (variable) thuộc Session object để giữ giá trị (hay nội dung, **value**) của hộp chữ đó và sẽ vận dụng các biến số (variables) đó trong bài tập kế.

Bài tập 3:

Mục đích:

Tạo 1 trang Web khác dùng Session object để truy tìm và sử dụng biến số của Session ở bài tập 2. **Nhớ đừng đóng (close) browser ta vừa dùng để hiển thị trang Session.aspx ở trên.**

Các bước thứ tự như sau:

1. Chạy ứng dụng Notepad và gõ nguồn mã sau:

```
<%@Page Language="VB" %>

<script runat="server">

sub Page_Load(obj as object,e as EventArgs)
lblWelcome.Text="Welcome back " & Session("Name ")
end sub

</script>

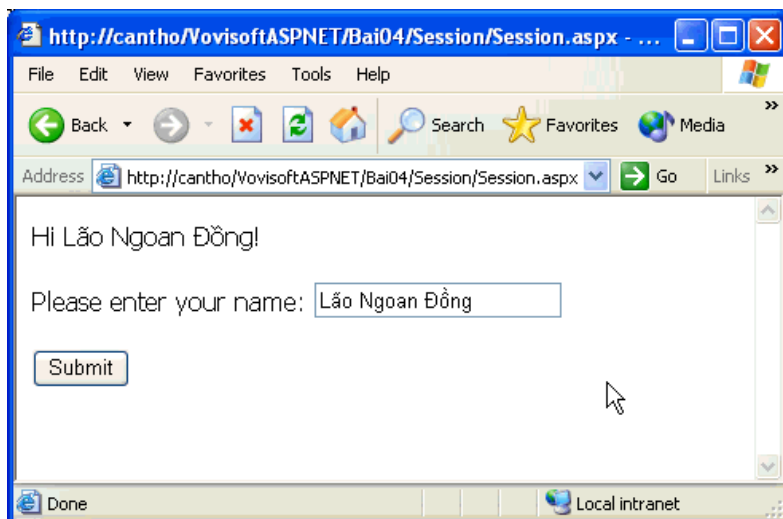
<html>
<body>

<form runat="server">
<asp:Label id="lblWelcome" runat="server"/>
</form>

</body>
</html>
```

2. Lưu trữ tập tin với tên SessionVariables.aspx ở folder 'D:\Net\Vovisoft ASPNET\Bai04\Session' và hiển thị trang Web với URL sau: **http://cantho/VovisoftASPNET/Bai04/Session/Session.aspx**

Ta thấy Session đã truy tìm được thông tin ở trang Web khác qua biến số (variables) Name mà ta cung cấp:



Phần Chú Thích:

Ta thử tìm hiểu xem làm thế nào ASP.NET có thể quản lý đối tượng (objects) Session? Thông thường, mỗi khi có người lướt mạng, Session sẽ được tạo ra cho người đó 1 instance, ấn định 1 lý lịch (Session ID) riêng biệt, duy nhất và lưu trữ ở máy vì tính của user dưới dạng **cookie**. Như vậy, khi user lướt từ trang này qua trang nọ, ASP.NET sẽ truy tìm và đọc lại giá trị của cookie để xác nhận user thuộc về session nào với mệnh lệnh sau:

```
Response.Write(Session.SessionID)
```

Session ID này dài 120-bit string, đủ bảo đảm tính chất duy nhất cho từng user. Khi session tới hạn kỳ, cookie sẽ được xóa sạch (trên thực tế, tình trạng này có thể được thay đổi và kiểm soát - tuy vậy ta không bàn chi tiết ở đây).

Kiểm soát (control) Session Object

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

- Timeout
- Abandon

Timeout dùng bố trí khoảng thời gian bao lâu 1 Session có thể ... 'ngồi chơi soi nước' hay 'thất nghiệp' trước khi ASP.NET 'tự' bỏ session đó, tỷ như có người lướt mạng thăm trang Web của ta rồi ... 'bỏ đi ăn trưa' hay ... 'ngồi chết trân' không nhấp mũi chuột bất cứ gì, session và các thông tin liên hệ sẽ ... 'tan biến' trong vòng 20 phút. 20 phút này là giá trị timeout mặc định (default) nếu dùng IIS Version 5.0, nếu muốn, ta có thể thay đổi giá trị timeout thành 60 phút chẳng hạn:

```
Session.Timeout = 60
```

Tuy vậy, ta nên cẩn thận khi thay đổi giá trị timeout này vì:

Mỗi user lướt mạng đều được gán hay đính kèm 1 đối tượng (Object) Session duy nhất lưu trữ trong memory của Server. Đặt thí dụ như nếu người ta mới chào hỏi chừng 1 phút rồi đã vội sang ... 'ngang' qua mạng khác, như vậy session mặc định (default) 20 phút của họ vẫn còn .. 'hấp hối' thêm 19 phút nữa mới ... 'ngừng thở', rất tốn memory. Nếu ta có chừng vài ngàn hay vài trăm ngàn user lướt mạng cùng 1 lúc, không khéo Server của ta phải ... 'dẹp tiệm, về hưu non'.

Mặt khác, nhất là đối với các mạng thương nghiệp (**e-commerce site**), nếu bố trí timeout quá ngắn, các sản phẩm được chọn mua và đặt trong **shopping cart** sẽ xóa sạch trước khi khách hàng tiến hành thủ tục trả tiền và như vậy gây trở ngại lớn lao cho thương vụ.

Một cách tổng quát, timeout 20 phút là lý tưởng. Tuy nhiên, ta có thể thay đổi giá trị tùy theo tính chất của mạng, đối với ngân hàng (secure banking web site), timeout có thể rất ngắn nhưng với các mạng thương nghiệp, timeout sẽ lâu hơn. Đôi lúc, ta phải phác thảo hay tính toán trước khả năng của Server cho phù hợp với tình hình mạng, đừng để ... 'nước đến chân mới nhảy'.

Abandon dùng để kết thúc 1 session ngay tức khắc. Thí dụ, sau khi user kiểm tra email (Web Email) xong và logout để người khác không thể dùng ... 'ké' lúc họ vắng mặt, ta có thể kết thúc session đó bằng cách:

```
Session.Abandon
```

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

Làm việc với Session

Session được vận dụng như Array. Ta có thể xuyên suốt (loop through) và vận dụng (manipulate) các biến số (variables) của Session.

Bài tập 4:

Mục đích:

Dùng đối tượng (Object) Session để bố trí vài biến số (variables) và dùng For ... Next để xuyên suốt (loop through) và vận dụng (manipulate) các biến số (variables) của Session.

Các bước thứ tự như sau:

1. Chạy ứng dụng Notepad và gõ nguồn mã sau:

```
<%@Page Language="VB" %>
<script runat="server">

sub Page_Load(obj as object,e as EventArgs)
    dim strVariable as string

    'set up some session variables
    Session("Name")="Vovisoft"
    Session("Course")="ASP.NET For Beginner Course"
    Session("AMessage")="Welcome to VoviLearned Society"
    Session("ASPNET")="Session Object"

    'Manipulate each variable in session
    for each strVariable in Session.Contents
        lblSession.Text += "<b>" & strVariable & "</b>:" & _
            Session(strVariable)&"<br>"
    next
end sub

</script>

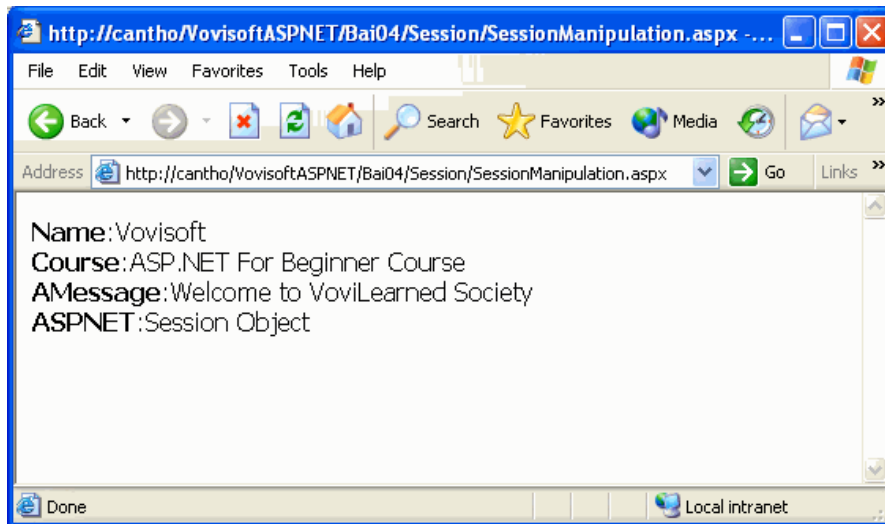
<html>
<body>

<form runat="server">
```

```
<asp:Label id="lblSession" runat="server"/>
</form>

</body>
</html>
```

2. Lưu trữ tập tin với tên SessionManipulation.aspx ở folder 'D:\Net\Vovisoft ASPNET\Bai04\Session' và hiển thị trang Web với URL sau: **http://cantho/VovisoftASPNET/Bai04/Session/Session.aspx**



Phần Chú Thích:

Khi ấn định giá trị cho các cặp biến số (variables) của Session, ta dùng kiểu **key/value** với key là argument riêng biệt và duy nhất để truy tìm giá trị tương ứng. Như vậy, ta chỉ cần gọi Session với **key** liên hệ trong **For ... Next** loop, tỷ như : key Name với giá trị Vovisoft, key Course với giá trị ASP.NET For Beginner Course, ... và giữ tất cả trong đặc tính (properties) Text của nhãn hiệu lblSession.

Sessions không Cookies

Thông thường, ASP.NET dùng Cookies để lưu trữ Session ID và thông tin về user. Nhưng, điều gì xảy ra nếu browser của user không yểm trợ cookies hoặc trong nhiều trường hợp, user không muốn (hay chấp nhận) yểm trợ cookies ?

ASP.NET dùng 1 phương pháp (method) khác gọi là **Cookie Munging** tiến hành việc bảo quản session không cookies. Phương pháp (method) này tự khởi động nếu ASP.NET phát hiện ra user không yểm trợ cookies. Giá phải trả là việc thông tin sẽ chậm trễ đôi chút vì ASP.NET phải mã hoá và giải mã (encode and decode) Session ID cùng với các thông tin liên hệ và đặt vào các trang Web trước khi gửi đi. Tiến trình thứ tự như sau:

- User yêu cầu hiển thị 1 trang Web
- ASP.NET sẽ dò (scan) từng nối (hyperlinks) trong mã HTML (HTML code)
- Cứ cuối mỗi nối như vậy, ASP.NET sẽ gắn thêm phần Session ID đã được mã hoá (encode), tỷ như ****
- Khi user nhấp 1 nối, ASP.NET sẽ giải mã (decode) và chuyển phần Session đó tới trang Web user yêu cầu
- Trang Web này vận dụng các biến số (variables) của Session và lập lại tiến trình để có thể giữ thông tin giữa các trang Web với nhau

Lưu ý:

Ta dùng đối tượng (Object) Session khi ta cần bảo trì một vài thông tin tỷ như username hay password. Trong trường hợp quá nhiều thông tin cần phải giữ, phương pháp (method) dùng cookies hay cơ sở dữ liệu (database) thích hợp và hiệu quả hơn cũng như tránh việc rỉ memory của Server.

Tóm tắt

Trong bài học hôm nay, ta tìm hiểu 1 số đối tượng (Object) cơ bản của ASP.NET. Bài kế sẽ bàn thêm về các đối tượng (Object) khác như [HttpCookies](#), [HttpApplication](#) và [HttpServerUtility](#) qua đó ta thu lượm kiến thức về vài hoạt động cơ bản của ASP.NET

Các đối tượng (Object) Request, Response và Session xem chừng giống như các đối tượng (Object) đó ở ASP cổ điển nhưng thật sự đã cung cấp nhiều công dụng, khái niệm và phương pháp làm việc (methodologies) khác hơn, vững vàng và hùng mạnh hơn ở sau hậu trường. Chẳng hạn, đối tượng (Object) Session có thể lưu trữ các biến số (variables) xuyên qua Server Farms cũng như khả năng lưu trữ các biến số (variables) này ngay cả khi Server ... 'tê' (crashes) hay ... 'sụp' mà ta sẽ bàn sau. Về ngôn ngữ lập trình VB.NET trong các bài tập, ta sẽ bàn 1 cách đại khái những gì liên hệ đến ASP.NET ở bài học số 6.

Download Source Code

[Nguồn mã bài tập 1](#)

[Nguồn mã bài tập 2](#)

[Nguồn mã bài tập 3](#)

[Nguồn mã bài tập 4](#)

Bài làm ở nhà

Câu hỏi 1: Static member là gì?

Câu hỏi 2: ASP.NET có yểm trợ **caching** hay không?

Bài làm 1: Dùng Web Matrix phát triển 1 trang ASP.NET với sự cố Load (Load event) và nhãn hiệu (Label) để hiển thị và thông báo cho user giờ giấc (current time) họ đang lướt mạng và Session ID. Nhớ kiểm tra đặc tính (properties) IsPostBack của Page object để kiểm soát và hiển thị thông tin lần đầu user viếng trang Web mà thôi, bạn cũng có thể vận dụng thêm công cụ Button và 1 thông báo (confirmation message) cho user khi họ logout.

Bài 05

Dùng ASP.NET Objects với VB.NET (Part II)

Take it easy with me, please
Touch me gently like a summer evening breeze
Take your time, make it slow
Andante, Andante - Just let the feeling grow
Make your finger soft and light
Let your body be the velvet of the night
Touch my soul, you know how
Andante, Andante - Go slowly with me now
Andante - ABBA

Trong phần thứ 2 của bài 'Dùng ASP.NET Objects với VB.NET', ta sẽ lần lượt tìm hiểu:

- HttpCookie Object
- HttpApplication Object
- HttpServerUtility Object

HttpCookie Object

Như đã trình bày **Cookie** chỉ là 1 tập tin nhỏ ở máy vi tính của user trong đó chứa mọi thông tin đặc trưng cho 1 mạng nào đó, tỷ như tên người sử dụng và mật mã (username and password) nhằm lưu trữ thông tin cá biệt liên quan đến từng user một khi user lướt mạng (customize a user's visit to a site).

Nhớ là Cookie chỉ chứa các kiểu dữ kiện đơn giản (simple data type), tỷ như strings, integers, floats, booleans, ... Một thí dụ phổ thông là nhiều mạng hiển thị các tiêu đề (headlines) cho phép user chọn những gì họ thích và chứa thông tin cá biệt đó trong cookies, khi user tái viếng thăm, mạng chỉ hiển thị những gì mà họ lựa chọn trước đây giống như mạng của chính họ phác thảo và phát triển.

HttpCookie object cung cấp các phương pháp (method) để tạo ra và vận dụng những cookies đó. Ta có thể dùng HttpCookie Object để kiểm tra các đặc tính (properties) của từng cookie.

Tuy nhiên, phương pháp (method) dùng **Request** và **Response** Objects là các phương pháp (method) thông dụng nhất để vận dụng cookie, cả hai đều có đặc tính (properties) Cookies mà giá trị của nó liên quan (it returns a reference to) đến HttpCookie Object.

Tạo ra Cookies

Ta dùng Response Object để tạo ra Cookies với 2 cách sau:

- Tạo nhiều Cookies, mỗi cookie kèm với một giá trị (gọi là Cookies đa dạng, đơn giá trị - multiple Cookies, each with a single value)
- Tạo một Cookie với nhiều cặp **key/value** (gọi là Cookie đơn dạng, đa giá trị)

Thí dụ:

```
'Kiểu Cookies đa dạng, đơn giá trị  
Response.Cookies("MyCookie1").Value = "Single Cookie Value 1"  
Response.Cookies("MyCookie2").Value = "Single Cookie Value 2"
```

```
'Kiểu Cookie đơn dạng, đa giá trị
Response.Cookies("MyASPNETPage").("Username") = "Nang Vu"
Response.Cookies("MyASPNETPage").("Password") = "TakeMeHome"
```

Ta nhận thấy ở kiểu Cookies đa dạng, đơn giá - mỗi cookie có tên khác nhau ("MyCookie1" và "MyCookie2") với giá trị được xác định dùng **.Value** = "...". Ta không nên dùng Response.Cookies("MyCookie1") mà không có thành phần .Value = "..." vì chỉ trả lại (return) 1 đối tượng (Object) HttpCookie Object, làm ta không thể bổ trí một giá trị nào cả.

Còn ở kiểu thứ nhì, ta dùng từng cặp key/value cho mỗi dạng và giá trị riêng biệt nhưng cookie thì chỉ một tên thôi (ở đây là "MyASPNETPage"). Ta cũng không cần phải nhắc tới value vì ASP.NET biết rõ ràng ta muốn gì khi xấp đặt một **key** (khóa) chuẩn bị lưu trữ một giá trị.

Nhớ dùng ... 'xào thập cẩm' 2 kiểu trên với nhau. **Ta không thể nào tạo ra một Cookie vừa có Value vừa có key.** Ông bà có dặn rồi "Lắm mồi, tôi ... nằm không" đấy.

Đáo hạn (Expires) Cookies

Giả như có người lướt mạng và ta tạo ra 1 cookie để lưu trữ thông tin về người ấy, nhưng nếu người ấy ... 'ra đi không hẹn ngày trở lại' thì không có lý do gì mà ta 'vẫn giữ mãi trong ... tim', à không, ... trong máy vi tính của họ. Tuy họ có thể dễ dàng tự xóa bỏ cookie, nhưng hay hơn nữa, ta có thể dùng đặc tính (properties) **Expires** để ... 'dùng gieo gánh nặng, nửa đường tội em' như sau:

```
'Kiểu ngày giờ đáo hạn
Response.Cookies("MyASPNETPage").Expires = DateTime.FromString("30/02/2003")
```

'hoặc là

```
'Kiểu khoảng thời gian đáo hạn
Response.Cookies("MyASPNETPage").Expires = DateTime.Now.AddMonths(1)
```

Kiểu đầu ra lệnh cho Cookie đáo hạn vào ngày 30 tháng Hai năm 2003 (ủa, chẳng biết ở xứ Congo có ngày 30 tháng Hai không nhỉ ?), còn kiểu thứ nhì đáo hạn một tháng kể từ khi nguồn mã được thi hành.

Đề ý, giá trị đáo hạn mặc định của cookies là 1000 phút (1,000 minutes). Điều này cũng giúp ta bảo trì và vận dụng các thông tin tạo ra bởi các **session** hiện hành (current sessions). Nhưng giá trị này thường được thay đổi vì thông thường, cookies dùng để lưu trữ các thông tin trong khoảng thời gian lâu dài hơn, tỷ như vài tuần, vài tháng hay cả năm không chừng.

Để **xóa sạch (delete) cookie** ở máy Client, ta đơn giản bổ trí giá trị của Expires thành **0** hay giá trị thời gian thuộc về quá khứ, Cookie sẽ biến mất khi user đóng (close) browser của họ.

HttpCookie Object còn vài đặc tính (properties) cần lưu ý như sau:

- Domain
- Path
- HasKeys
- Secure

Domain dùng để giới hạn việc sử dụng cookies ở một domain ta chỉ định, tỷ như www.myserver.com

Path dùng tương tự như Domain, nhưng giới hạn việc sử dụng cookies ở **path** chỉ định nào đó trong Server của ta.

HasKeys báo cho ta biết Cookie dùng kiểu 'đa dạng, đơn giá trị' với các cặp key/value.

Secure báo cho ASP.NET biết nên chuyển cookie 1 cách an toàn hay không và thường chỉ xảy ra ở trên HTTPS protocol. Giá trị mặc định (default) của Secure là False.

Liên hệ (Access) với Cookies

Client browser gửi tất cả thông tin ở Cookie tới Server mỗi khi yêu cầu hiển thị (display) trang Web. Ta có thể dùng đối tượng (Object) Request to thu thập các thông tin đó. Ta dùng cùng 1 cú pháp như ở phần 'Tạo ra Cookies' để liên hệ với cookie, tỷ như thu thập giá trị và viết trả lại Client browser như sau:

'Cho kiểu Cookies đa dạng, đơn giá trị

Response.Write (Request.Cookies("MyCookie1").Value)

Response.Write (Request.Cookies("MyCookie2").Value)

'Cho kiểu Cookie đơn dạng, đa giá trị

Response.Write (Response.Cookies("MyASPNETPage").("Username"))

Response.Write (Response.Cookies("MyASPNETPage").("Password"))

Bài tập 1:

Mục đích:

Trong bài tập này, ta sẽ xây dựng một trang ASP.NET dùng Cookies để lưu trữ 1 số thông tin ở Client browser và sau đó liên hệ, vận dụng và hiển thị (display) các thông tin đó trên cùng 1 trang Web.

Các bước thứ tự như sau:

1. Chạy ứng dụng Notepad và gõ hàng mã sau:

```
<%@ Page Language="VB" %>
```

```
<script runat="server">
```

```
sub Page_Load(obj as object, e as EventArgs)
```

```
    dim strVariable as string
```

```
    'set up some cookie variables
```

```
    Response.Cookies("MyASPNETPage")("Username") = "Nang Vu"
```

```
    Response.Cookies("MyASPNETPage")("Password") = "TakeMeHome"
```

```
    Response.Cookies("MyASPNETPage")("Preference") = "800x640"
```

```
    Response.Cookies("MyASPNETPage")("UserAgent") = _
```

```
        Request.ServerVariables("HTTP_USER_AGENT")
```

```
    for each strVariable in Response.Cookies("MyASPNETPage").Values
```

```
        lblCookies.Text += "<b>" & strVariable & "</b>: " & _
```

```
            Request.Cookies("MyASPNETPage")(strVariable) & "<br>"
```

```
    next
```

```
end sub
```

```
</script>
```

```
<html>
```

```
<body>
```

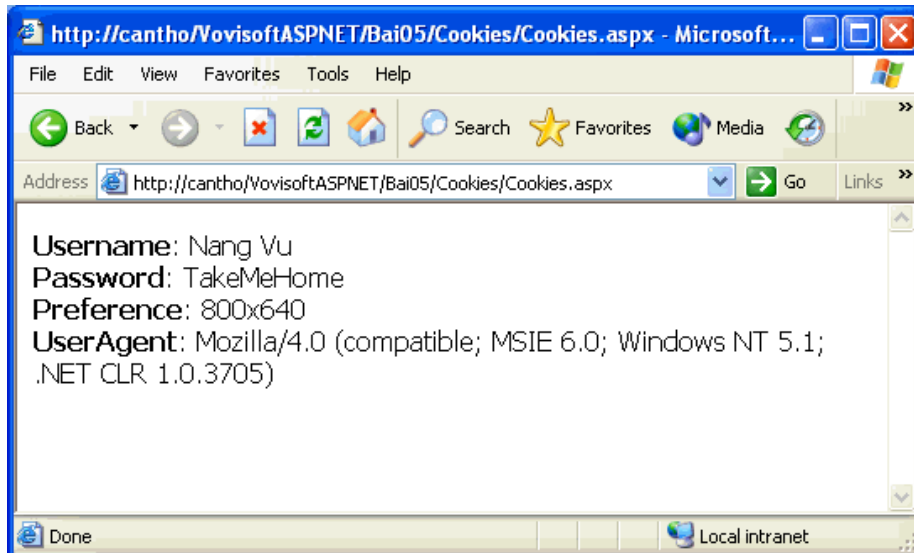
```

<form runat="server">
  <asp:Label id="lblCookies" runat="server"/>
</form>

</body>
</html>

```

2. Lưu trữ tập tin với tên Cookies.aspx ở folder 'D:\Net\Vovisoft ASPNET\Bai05\Cookies' và chạy IE với URL như sau: <http://cantho/VovisoftASPNET/Bai05/Cookies/Cookies.aspx>



Phần chú thích:

```

'set up some cookie variables
Response.Cookies("MyASPNETPage")("Username") = "Nang Vu"
Response.Cookies("MyASPNETPage")("Password") = "TakeMeHome"
Response.Cookies("MyASPNETPage")("Preference") = "800x640"
Response.Cookies("MyASPNETPage")("UserAgent") = _
  Request.ServerVariables("HTTP_USER_AGENT")

```

Ở trên, ta đã bàn qua về việc bố trí 'key/value' cho Cookies 'MyASPNETPage' tỷ như key tên "Username" cho giá trị "Nang Vu". Ở hàng mã cuối, ta dùng bộ sưu tập (collection) **Request.ServerVariables** để thu lượm giá trị của một trong các thành viên của nó là **HTTP_USER_AGENT**.

Thành viên **HTTP_USER_AGENT** này cho ta biết phiên bản browser của máy Client.

```

for each strVariable in Response.Cookies("MyASPNETPage").Values
  lblCookies.Text += "<b>" & strVariable & "</b>: " & _
    Request.Cookies("MyASPNETPage")(strVariable) & "<br>"
next

```

Ở đây, ta dùng For Each ... Next loop để tìm lại giá trị của từng 'key' trong cookie 'MyASPNETPage' và hiển thị các cặp key/value thành từng hàng một nhờ ở HTML tag '
'.

Về việc dùng For Each ... Next loop, ta sẽ tham khảo riêng ở bài 6 '**.NET Framework và VB.NET**'.

HttpApplication Object

HttpApplication Object biểu hiện cho ứng dụng (application) ASP.NET . Ở phần mục này, ta chỉ sơ lược về HttpApplication Object và sẽ đào sâu thêm chi tiết ở các bài sau.

Xin nhắc lại ở đây, ASP.NET định nghĩa **ứng dụng (application)** bao gồm tất cả mọi tập tin kể cả các sub folder và các tập tin trong đó ở 1 **virtual directory** tỷ như **VovisoftASPNET** mà ta vẫn thường sử dụng.

Cũng tương tự như đối tượng (Object) Response, ASP.NET tạo ra 1 HttpApplication Object gọi là Application chỉ khi nào ứng dụng (application) của ta khởi động - nghĩa là khi có user yêu cầu tham khảo trang Web trong site của ta lần đầu tiên. Lưu ý chỉ có duy nhất một Application object được tạo ra cho toàn bộ ứng dụng (application) mà thôi, không như Session object được tạo ra riêng biệt cho từng user một. Tuy vậy, HttpApplication Object giống Session Object ở chỗ: HttpApplication Object cũng được dùng để lưu trữ các biến số và các đối tượng (Object) liên hệ. Các biến số và các đối tượng này có hiệu lực (available) cho toàn bộ ứng dụng (application) chứ không cá biệt cho một ai.

Thí dụ, ta muốn hiển thị (display) 1 'footer' hay 'disclaim' ở cuối mỗi trang Web, ta lưu trữ biến số (variable) tỷ như 'Disclaimer' của ứng dụng (application) như sau:

```
Application("Disclaimer") = "Copyright 2002"
```

và sau đó, ở mỗi trang Web ta thu lượm lại:

```
Response.Write(Application("Disclaimer"))
```

Việc dùng HttpApplication Object có thể giúp ta tránh rỉ memory vì chỉ lưu trữ 1 lần nhưng dùng ở mọi trang Web. Ngược lại, nếu ta lưu trữ cũng cùng một thông tin với Session object, ASP.NET sẽ lưu trữ riêng biệt cho từng user và ta có thể hiểu được việc gì sẽ xảy ra nếu hàng trăm hoặc hàng nghìn (hay hàng triệu) user ... 'đến hẹn lại lên' cùng một lúc.

HttpServerUtility Object

HttpServerUtility Object được dùng để cung cấp nhiều 'helper methods' giúp giải quyết tiến trình yêu cầu (used in processing requests) của user. Ta có thể dùng object này dưới tên gọi là **Server** để liên hệ các thành viên của HttpServerUtility Object. Các 'helper methods' có thể kể ra như sau:

- Redirecting Users
- Formatting Strings
- Controlling Scripts
- Creating Objects

Redirecting Users

Ta đã làm quen với cách chuyển user qua 1 trang Web khác với phương pháp (method) **Response.Redirect**, phương pháp (method) này gửi kèm với 1 dạng thông tin HTTP tới browser chỉ thị cho biết việc chuyển như thế nào. Việc chuyển này đòi hỏi 1 đường vòng (round trip) không cần thiết vì user phải thăm trang Web của ta trước được chuyển đi.

ASP.NET có thể chuyển user trực tiếp qua trang Web khác với phương pháp (method) **Execute** và **Transfer** thuộc HttpServerUtility Object. Thật vậy, Server.Transfer thi hành việc chuyển qua trang Web khác 1 cách đơn giản mà không đòi hỏi bất cứ thông tin phải gửi tới browser, công việc này tiến hành sau hậu trường và user không hề hay biết.

Thí dụ sau đây cho thấy nguồn mã kiểm tra mật mã của user và chuyển user qua 1 trang Web khác 1 cách trực tiếp (và ngay tức khắc):


```
if (strPassword == "TakeMeHome")
    Server.Transfer("products.aspx")
end if
```

Formatting Strings

Có 4 phương pháp (method) cơ bản cần để ý để cấu tạo hay hình thành 1 hàng chữ (format a string):

- Server.HtmlEncode
- Server.UrlEncode
- Server.HtmlDecode
- Server.UrlDecode
- Server.MapPath

HTMLEncode

Mỗi khi ta gửi kết quả (output) tới browser, ASP.NET tự động thông dịch nguồn mã thành dạng HTML tương đương, tỷ như để thêm vào 1 'line break' sau:

```
Response.Write("<br>")
```

Tuy nhiên, một đôi khi ta muốn hiển thị (display) chính hàng chữ
 thay vì 1 'line break' trên máy Client, ta có thể dùng phương pháp (method) **HTMLEncode** của đối tượng (Object) HttpServerUtility Object.

Thí dụ sau hiển thị (display) kết quả của các ký hiệu **
** trên máy Client và user sẽ thấy hàng chữ
 thật sự:

```
Response.Write(Server.HtmlEncode("<br>"))
```

Lưu ý: **<** là nguồn mã của HTML cho ký hiệu <, **>** cho ký hiệu >.

URLEncode

URLEncode dùng như HTMLEncode nhưng hình thành (format) hàng chữ với các điều lệ (rules) dành cho URL, tỷ như dấu ampersand (&) và dấu chấm hỏi (?) mang một ý nghĩa đặc trưng trong các URL, do đó Server.URLEncode cần chuyển các ký hiệu như vậy qua dạng dùng cho phiên bản URL (**URL-encoded version**). Ta sẽ thấy sự quan trọng như thế nào khi xấp đặt 1 hàng chữ dùng các ký hiệu đó dính kèm trong 1 **querystring**.

HTMLDecode, URLDecode

Các phương pháp (method) HTMLDecode và URLDecode được dùng như các HTMLEncode và URLEncode tương ứng nhưng thực hiện 1 tiến trình đảo ngược lại, nghĩa là chuyển (dịch, translate) các chuỗi ký hiệu mã hóa (encoded character sequences) trở lại dạng nguyên thủy, tỷ như chuyển **<** trở lại ký hiệu < chẳng hạn.

MapPath

Phương pháp (method) **MapPath** không dùng để hình thành 1 hàng chữ nhưng giúp ta xác định rõ ràng các string ta cần, nhất là trong việc dịch (hay chuyển) 1 virtual directory path trở lại dạng nguyên thủy dùng ở đĩa cứng trong Server (**translate a virtual path to a physical path on a server**) tỷ như chuyển:

```
Server.MapPath("/VovisoftASPNET/Bai05")
```

Điều này rất tiện lợi khi ta cần phải biết 1 physical path, chẳng hạn như khi cần phải đọc và viết các tập tin ở Server. Ở bài này, ta sẽ không tham khảo về việc đọc và viết các tập tin ở Server.

Controlling Scripts

Việc kiểm soát sự thi hành các ASP.NET script, HttpServerUtility object có phương pháp (method) **ScriptTimeout**. ScriptTimeout bố trí khoảng thời gian Sever phải chờ đợi trước kết thúc 1 script. Ta có thể dùng mệnh lệnh sau để bố trí khoảng thời gian chờ đợi là 90 giây như sau:

```
Server.ScriptTimeout = 90
```

Việc kiểm soát sự thi hành các ASP.NET script cũng cần thiết khi nguồn mã ta có lỗi (bug) tỷ như **infinity loop** có thể làm té (crash) Sever của ta, ScriptTimeout có thể được bố trí để kết thúc các script chạy quá lâu hoặc không cư xử đúng phép tắc hoặc không đúng yêu cầu như đã hoạch định trước.

Tuy vậy, ta cũng để ý, khi bố trí ScriptTimeout quá ngắn, có thể gây trở ngại không cần thiết cho các nguồn mã cần thi hành trong 1 khoảng thời gian dài, lâu hơn giá trị hiện hành của ScriptTimeout.

Creating Objects

CreateObject được dùng để tạo ra 1 đối tượng (Object) định hình (instantiates a COM object) 1 đối tượng (Object) COM xác minh với **progid**. Ta sẽ tham khảo chi tiết các dùng phương pháp (method) Creating Object này ở các bài sau.

Tóm tắt

Trong OOP, đối tượng (Object) dùng đặc tính (properties) và phương pháp (method) để tự diễn tả hoặc cho biết mình là gì. Thông thường đối tượng (Object) cần định hình (instances) trước khi dùng các đặc tính (properties) và phương pháp (method) đó, ngoại trừ static members.

Ta cũng sơ lược qua về HttpCookies dùng để cung cấp 1 cơ chế đọc và viết thông tin vào cookies, về HttpApplication đại diện cho ứng dụng và về HttpServerUtility với các phương pháp (method) giúp xử lý các yêu cầu của Client. Ta nhận thấy ngôn ngữ lập trình VB.NET được sử dụng để viết các nguồn mã trong phần Script cho các trang ASP.NET, do đó việc tìm hiểu ngôn ngữ lập trình VB.NET là việc cần thiết và quan trọng đối với những ai muốn phát triển mạng với ASP.NET. Trong bài tới, ta sẽ bàn sơ lược về .NET Framework, các classes và ngôn ngữ lập trình VB.NET cùng cú pháp cơ bản có liên hệ ít nhiều đến các trang ASP.NET

Download Source Code

[Nguồn mã bài tập 1](#)

Bài làm ở nhà

Câu hỏi 1: Kể vài đặc tính (properties) và ứng dụng dùng HttpCookies?

Bài làm 1: Phát triển vài trang Web dùng `HttpApplication` object để hiển thị (display) cùng 1 hàng chữ chẳng hạn như 'Vovisoft @ 1998 - 2002 All rights reserved'.

Bài làm 2: Phát triển 1 trang Web dùng `HttpServerUtility` object để cấu tạo (hay hình thành) các hàng chữ theo ý bạn với mục đích làm quen các phương pháp (method) sau đây:

- `Server.HtmlEncode`
- `Server.UrlEncode`
- `Server.HtmlDecode`
- `Server.HtmlDecode`

Bài 06

ASP.NET và VB.NET

Cùng góp bàn tay thương yêu nhau rồi
Ngô khoai hai mùa ngát một niềm vui chung vui
Cho thơm hương đời lúa vàng tình ơ
Ngày mai hạnh phúc nơi nơi reo cười
Quê hương thôi đau sầu ngăn sông núi cách chia
Ta đem yêu thương về cho phương Bắc
Khúc Hát An Tĩnh - Xuân Tiên

Ta đã biết ASP.NET nằm trong cấu trúc nền của .NET framework và dùng .NET programming language tỷ như VB.NET (một trong 25 ngôn ngữ lập trình .NET hiện nay) để phát triển trang Web, do đó ta cũng cần biết sơ lược về .NET framework cùng ngôn ngữ lập trình VB.NET dùng cho các trang ASP.NET.

Chi tiết phương pháp lập trình theo khuynh hướng đối tượng (Object Oriented Programming) và VB.NET, xin tham khảo các bài viết về [OOP và Visual Basic.NET](#) do **thầy Lê Đức Hồng** soạn và trình bày.

Trong bài 'ASP.NET và VB.NET', ta sẽ lần lượt tìm hiểu:

- Giới thiệu tổng quát về .NET Framework
- Sơ lược về ngôn ngữ lập trình VB.NET và cú pháp
- Functions và SubRoutines
- Classes
- Phương pháp (method) lập trình tổng quát

.NET Framework

Một cách tổng quát, .NET Framework là kiểu mẫu lập trình cách mạng cho tất cả những gì liên quan đến nền Windows.

Nhìn lại hơn 10 năm, Microsoft đã từng bước một cải tiến nền Windows (Windows platform) hết sức nghiêm túc, đạt nhiều thành công rực rỡ với những API (Application Programming Interface) cho mọi ứng dụng, với hàng loạt công cụ phát triển và lập trình (developer tools) hết sức thuận lợi cho các Kỹ Sư Tin Học cũng như cho những ai yêu thích các sản phẩm của Microsoft. Ngoài ra, ta còn chứng kiến sự xuất hiện của COM, rồi DCOM, COM+ với mục tiêu tái sử dụng các nguồn mã hay nhu liệu tương ứng. Thêm nữa, nào là ODBC (Open Database Connection), DAO, OLEDB tới ADO (Active Data Object) với rất nhiều phiên bản (versions) cho việc nối kết các cơ sở dữ liệu (Database) gồm đủ loại đủ cỡ. Về phương diện mạng năng động (dynamic web sites), ta có nhiều phiên bản ASP cổ điển hay về ngôn ngữ lập trình với C++, J++, Visual Basic cùng các ngôn ngữ lập trình scripting như VBScript, ...

Nhưng ...

Mặc dù, các công cụ cũng như các ngôn ngữ lập trình với nhiều phiên bản khác nhau như vừa nêu trên, yểm trợ những gì công nghệ thông tin đang đòi hỏi - chúng càng lúc càng trở nên phức tạp và rắc rối cho các Chuyên Gia hay Kỹ Sư Tin Học, không những không theo kịp đà tiến triển hiện nay mà còn không tạo được một nền tảng hùng mạnh cho việc phát triển hay đáp ứng sự thay đổi nhanh chóng của công nghệ thông tin cho kỹ nghệ, thương mại, đời sống trong tương lai.

Chẳng hạn một Kỹ Sư Tin Học giàu kinh nghiệm muốn học về COM - 1 phương pháp hết sức hùng mạnh để gói ghém các nguồn mã cho việc tái sử dụng - cần tối thiểu 6 tháng hay cả năm để làm quen hay nắm vững. Còn nếu muốn học về DCOM hay COM+, ta phải biết COM trước. Đó cũng là lý do tại sao

những Kỹ Sư Tin Học chán nản vì quá nhiều vá vúi hoặc nhiều thêm thắt cho các ứng dụng mà ngay từ thừa ... 'hồng hoang' đã phác thảo một cách nghèo nàn ảnh hưởng từ hoàn cảnh xã hội và công nghệ lúc bấy giờ.

Microsoft biết điều đó nên xóa ... 'cờ làm lại từ đầu'. Việc đầu tư vào .NET Framework với mọi tài nguyên mà Microsoft (hay ... trên trái đất này) có được, cho thấy sự xuất hiện của .NET Framework không phải là chuyện tầm ... 'cỏ' đâu, ủa quên, ... 'tầm thường' đâu.

.NET Framework phác thảo bắt đầu từ con số 0 với tham vọng bao trùm đủ mọi thứ và cũng không quên ... nâng đỡ 'bạn xưa' nên các nguồn mã ... 'cũ kỹ' của ta vẫn được yểm trợ. Không những thế mà Microsoft đã và còn lắng nghe mọi nguyện vọng của .. các Kỹ Sư Tin Học hay của ta trong việc kế hoạch và phác thảo cấu trúc nền .NET qua đó đem lại nhiều lợi ích thiết thực cho người lập trình hay phát triển và tạo vận hội mới phát triển nền Công Nghệ Tin Học.

Việc tường trình về .NET Framework làm ta nhớ lại sự xuất hiện của Linux trước đây. Nhìn lại 10 năm vừa qua, Linus Torvalds (cha đẻ Linux) cũng đã quá chán nản với sự giới hạn và khuyết điểm của Unix mà viết lại Operating System này từ con số 0 thay vì cố gắng sửa đổi hay vá vúi. Việc làm lại từ đầu đó chỉ với mục đích là theo kịp các đòi hỏi, nếu không muốn nói là vạch hướng đi cho tương lai Tin Học. Nếu ta ví sự phổ thông và nổi tiếng của nền Windows với Microsoft (Bill Gates) như là phái Thiếu Lâm thì Linux sẽ là 'Thái Cực Quyền' của phái Võ Đang với Trương Tam Phong (Linus Torvalds) và mặc dù chỉ chiếm 0.25% thị trường desktop trên thế giới (trích Next Handbooks - Operating in Linux by Paul Robinson and Dan Corkery - 2002) nhưng cũng đã góp phần làm cho Công Nghệ Thông Tin thêm phần đặc sắc và muôn màu muôn vẻ. Mỗi system đều có cái hay và dở khác nhau khó lòng so sánh tỷ như tuy Linux 'free OS' (và Open Source) ràng buộc bởi GPL (GNU Public Licence by Richard Stallman) nhưng gần đây lại chia thành nhiều phe nhóm - Nam Tông (United Linux) gồm có SuSe, Caldera, Connectiva and Turbo Linux, ... và Bắc Tông (Unbreakable Linux) với Red Hat, Oracle, ... nên đủ thứ phiên bản Linux khác nhau có thể làm ta ... 'ngất ngư' chẳng biết theo ai, còn Windows (Close Source) với .NET Framework rất hùng mạnh nhưng ... lại 'tốn tiền', nếu có nhiều ứng dụng 'free' tỷ như Web Matrix ... 'cho không biếu không' thì tốt cho cộng đồng .NET biết bao nhiêu.

Tuy nhiên, ở đây không phải diễn đàn để ... 'Hoa Sơn Luận Kiếm' giữa Windows, Linux hoặc MAC OS nên ta gác lại để chỉ bàn về .NET Framework và các lợi ích của nó. Thật sự, mỗi nền đều có cái hay của nó và là những bông hoa đẹp, hiếm quý trong vườn Tin Học mà ta luôn luôn trân trọng - tất cả đều là sản phẩm trí óc tuyệt vời của con người, nếu bạn có khả năng và thời giờ thì ... học càng nhiều càng tốt.

Lợi ích của .NET Framework

Ở đây, ta không kể xiết hay đào sâu chi tiết lợi ích của .NET Framework mà chỉ tóm lược vài điểm chính yếu sau:

- Mọi chuyện ... 'trên trời dưới đất' mà ta muốn thực hiện trên nền Windows, tỷ như data access, windowing, nối mạng hay ngay cả mọi công dụng đa dạng của Win32 API (Application Programming Interface) đều có thể vận dụng dễ dàng qua kiểu mẫu đối tượng (objects) rất đơn giản (simple object model).
- Ngôn ngữ lập trình VB.NET đã được hiện đại hóa, bao gồm nhiều classes và mọi đặc trưng (features) của 1 ngôn ngữ lập trình kiểu OOP, không thua kém gì C++, J++ hay C#, ...
- Việc quản lý memory được nâng cấp và tinh vi hơn nhằm bảo đảm các ứng dụng bị té hay cư xử tệ bạc (badly behaved component or application) không ảnh hưởng gì đến các ứng dụng khác.
- ASP.NET được dùng để thay thế ASP, đồng thời cung cấp các trang Web được biên dịch giúp tiến trình xử lý các yêu cầu từ Client browser hiệu quả hơn. Hơn nữa, còn bao gồm nhiều thành phần soạn sẵn (pre-written components) gọi là Server Control dùng trong các HTML Form và giao diện (user interface) làm việc phát triển mạng thêm dễ dàng và đầy hứng thú.
- Các ngôn ngữ lập trình được phác thảo để làm việc gần nhau hơn, do đó nguồn mã của VB.NET, C++, C#, ... có thể sử dụng trộn lẫn với nhau rất thoải mái, tỷ như ta có thể viết mã cho 1 class

với VB.NET rồi kế thừa 1 class khác mà mã là C# hay C++, ... sau đó vẫn 'debug' ngon lành giữa các ngôn ngữ lập trình khác nhau đó.

- Thành phần (**components**) được gói kỹ trong 1 đơn vị gọi là **assembly** có thể tự xác minh lý lịch và công dụng làm việc bố trí hay triển khai rất dễ dàng.

Phương pháp làm việc của .NET Framework

Điều kỳ thú nhất trong cấu trúc .NET Framework là các nguồn mã của VB.NET hay C# không biên dịch thành mã thi hành gốc (**native executable code**) mà lại qua trung gian một ngôn ngữ khác gọi là IL (Intermediate Language) trước khi chạy thật sự. Nguồn mã có thể biên dịch thành IL đó còn được gọi là managed code, điều này khiến cho các ngôn ngữ lập trình của .NET hoạt động (hay tác động) qua lại (hỗ trợ - interoperation) với nhau, cho phép ta vận dụng mọi đặc trưng của .NET mà không cần phải viết lại các nguồn mã dùng ngôn ngữ lập trình khác.

Nguyên tắc của IL cũng tương tự như Java, nhưng khác ở chỗ Java là cross-platform independence còn **.NET là cross-language independence**. Cũng cần phải nhắc ở đây, Microsoft vẫn mở rộng vòng tay cho việc phát triển .NET trên các nền (platform) khác trong tương lai.

Sơ Lược về VB.NET

VB.NET được dùng để tạo ra các trang Web đầy năng động với tất cả khả năng của ASP.NET (hay của .NET Framework). Ở đây, ta tìm hiểu tổng quát về cú pháp, cấu trúc và phương pháp lập trình với VB.NET có liên hệ ít nhiều đến các trang ASP.NET và hy vọng gặt hái được vài khái niệm cơ bản để việc lập trình xuống sẽ hơn.

Phần này, ta sẽ sơ lược qua các vấn đề sau:

- Đại khái về VB.NET
- Variables và Arrays
- Operators
- Các điều kiện (conditional), looping và branching logic
- Functions và Subroutines
- Event Handler
- Classes
- Vài ứng dụng ích lợi thường dùng

Đại khái về VB.NET

VB.NET là một trong 25 ngôn ngữ lập trình của .NET được yểm trợ bởi .NET Framework và CLR. VB.NET khác hẳn VB6 và thật sự chuyển mình thành 1 ngôn ngữ lập trình OOL chính yếu không khác gì với mọi ngôn ngữ lập trình khác tỷ như C#, C++, J++, ... trong môi trường xây dựng đủ mọi ứng dụng cho nền Windows và quan trọng hơn cả là việc học VB.NET dễ dàng hơn nhiều (its learning curve is not very steep). Do đó, VB.NET được chọn là ngôn ngữ lập trình cho các trang ASP.NET của khóa tự học này. Nếu bạn thích, bạn có thể dùng C# hay C++ thay vì VB.NET cho mọi bài tập trong khóa, chuyện đổi qua đổi lại giữa các ngôn ngữ lập trình chỉ là chuyện ... nhỏ, vì sự khác biệt phần lớn là về cú pháp (syntax) chứ nguyên tắc và cấu trúc lập trình thì giống y chang. Nhớ là dù ta dùng bất cứ ngôn ngữ lập trình nào, khi biên dịch vẫn phải qua ngôn ngữ trung gian Intermediate Language (IL) và quản lý bởi CLR.

Variables và Arrays

Variables

Variables là từ tổng quát gọi là **biến số** dùng lưu trữ dữ kiện (data) trong bộ nhớ (memory) của máy vi tính. Ta phải tuyên bố (declare) biến số trước khi dùng với keyword **Dim** tỹ như declare myVariable với loại (**Data Type**) String:

'Kiểu **explicit declaration** - VB.NET được báo cho biết ta muốn lưu trữ 1 string trong memory location đó
Dim myVariable As String

Data Types chia ra làm 5 **categories** gồm 10 loại cơ bản gọi là **primitive types**:

Type	Category	Description
Byte	Integers	1 byte (được biết như System.Int)
Short	Integers	2 bytes (System.Int16)
Integer	Integers	4 bytes (System.Int32)
Long	Integers	8 bytes (System.Int64)
Single	Floating-ppints	4 bytes với decimal point (System.Single)
Double	Floating-ppints	8 bytes (System.Double)
Decimal	Floating-ppints	12 bytes (System.Decimal)
Char	String	single Unicode character (System.Char)
Date	Dates	ngày giờ (Ssytem.DateTime)
Boolean	Boolean	Có/Không hay Đúng/Sai , True/False (System.Boolean)

Arrays

Arrays là 1 tập hợp các biến số được liên hệ riêng biệt qua chỉ số (**index**) của Arrays.

Nhớ là Arrays dùng trong VB.NET bắt đầu với index bằng số 0 - nghĩa là món hàng (item) đầu tiên được lưu trữ ở **index 0**, từ đó, suy ra chỉ số (index) món hàng sau cùng sẽ là tổng số các món hàng trừ đi một.

Mọi biến số trong Array phải cùng loại dữ kiện (same data type), không thể trộn lẫn nhiều loại khác nhau.

Array được tuyên bố (declare) như thí dụ sau đây:

'Tuyên bố array gồm 9 phần tử (hay thành phần - elements) thuộc loại Integer
Dim myArray(9) As Integer

'Tuyên bố array gồm 12 phần tử với giá trị mặc định (default) thuộc loại String
**Dim yourArray() As String = { "Tý", "Sửu", "Dần", "Mão", "Thìn", "Tỵ", _
"Ngọ", "Mùi", "Thân", "Dậu", "Tuất", "Hợi" }**

Operators (Ký hiệu Toán)

Operators là các ký hiệu dùng để thi hành 1 công việc thuộc phạm vi Toán Học, tỹ như dấu = dùng để ấn định (assign) 1 giá trị chẳng hạn như:

strSkills = "Thái Cực Quyền"

Để dễ dàng trong việc vận dụng các dấu Toán Học này, say đây là bảng liệt kê:

Công dụng (Function)	Operators (Các dấu Toán Học)
Exponentiation	^
Unary negation (tỹ như -9)	+, -
Multiplication, division	*, \
Division by (tỹ như 6/2 = 3)	/
Modulus (tỹ như 6 Mod 4 = 2)	Mod
Addition, Substraction	+, -
Bitwise NOT, AND, OR và XOR	BitNot, BitAnd, BitOr, BitXor
Concatenation (for string)	&, +
Equal to, not equal to, less than, greater than	=, <>, <, >
Less than or equal to, greater than or equal to	<=, >=
Relational	TypeOf ... Is, Is, Like
Assigment	=, ^=, *=, /=, +=, -=, &=

Điều kiện (conditional), looping và branching logic

Ta sẽ tham khảo cú pháp 3 logic sau:

- Conditional Logic
- Looping Logic
- Branching Logic

Conditional Logic

Conditional Logic cho phép ta chỉ định nguồn mã nào được thi hành tùy theo điều kiện đặt ra có phù hợp hay không. Có nhiều phương pháp (method) để quản lý conditional logic như sau:

- Phương pháp (method) dùng **If** statements
- Phương pháp (method) dùng **Case** statements

If statements

Tổng cộng 3 kiểu cú pháp như sau:

If (condition) Then

(your code)

...

...

End If

Như vậy, nếu điều kiện được thỏa mãn (condition = True), nguồn mã giữa If và End If sẽ được thi hành hoặc:

If (condition) Then

(your code for condition = True)

...

...

Else

(your code for condition = False)

...

...

End If

để thi hành nguồn mã khi condition = True hoặc False và

If (condition 1) Then

(your code for condition 1 = True)

...

...

ElseIf (condition 2) Then

(your code for condition 2 = True)

...

...

Else

(ngoài ra, thi hành code ở đây - your code for condition 1 and condition 2 = False)

...

...

End If

để thi hành nguồn mã khi condition 1 = True hoặc condition 2 = True hoặc khi cả hai condition 1 và condition 2 = False.

Case statements

Case statement thường gọi là **Select** statement giống như trường hợp If với nhiều ElseIf nhưng Case chỉ kiểm tra một biến số và tùy theo giá trị của biến số mà đáp ứng sao cho thích hợp. Cú pháp như sau:

```
Select Case variable
  Case option 1
    Code for option 1
  Case option 2
    Code for option 2
  Case Else
    Code
End Select
```

Looping Logic

Looping logic cho phép ta tái thi hành (hay lập đi lập lại) 1 công việc nào đó cho tới khi thoả mãn điều kiện đã định trước. Kiểu này gồm có 3 loại:

- While
- Do
- For

While Loops

Rất tiện lợi trong trường hợp ta không biết trước phải lập đi lập lại công việc bao nhiêu lần. Như vậy, kiểu loop này cơ bản dựa trên biểu thức có điều kiện (**conditional expressions**) và loop tái thi hành cho đến khi điều kiện định trước trở thành False. Cú pháp của 1 While loop như sau:

```
While condition
  Your Code
End While
```

Thí dụ ta muốn bố trí 1 máy đếm (counter) từ 1 đến 9 và hiển thị (display) kết quả bằng số ở browser:

```
'Bố trí counter loại Integer với giá trị 1
Dim intCounter As Integer = 1
```

'Bố trí While loop và hiển thị (display) giá trị của counter ở browser

```
While intCounter < 10
  Response.Write(intCounter & "<br>")
  intCounter += 1
End While
```

Ta nhận thấy khi giá trị của intCounter bằng 9, loop sẽ hiển thị (display) số 9 và sau đó cộng 1 vào intCounter thành ra 10 sẽ khiến cho điều kiện intCounter < 10 sẽ trở thành False, mã sẽ nhảy ra (exit) khỏi loop (loop exit), do đó ta chỉ thấy browser hiển thị (display) các số từ 1 đến 9 mà thôi.

Do Loops

Do loop cũng tương tự như while loop, chỉ khác ở chỗ Do loop thi hành công việc trước rồi mới kiểm tra điều kiện xem có phù hợp không? Kiểu này có thể gọi là kiểu ... 'tiền trạm hậu tẩu' khác với While loop là ... 'tiền tẩu hậu trạm' (?). Cú pháp như sau:

```
'Bố trí counter loại Integer với giá trị 1
Dim intCounter As Integer = 1
```

'Bố trí Do loop và hiển thị (display) giá trị của counter ở browser

```
Do
  Response.Write(intCounter & "<br>")
  intCounter += 1
```

Loop While intCounter < 10

Lần này có sự khác biệt so với While loop vì Do loop hiển thị (display) giá trị của intCounter trước, cộng thêm 1 rồi mới kiểm tra điều kiện. Ta phải lưu ý thứ tự thi hành trong trường hợp này, tỷ như ta bố trí giá trị của intCounter = 10 chẳng hạn, ta thấy browser sẽ hiển thị (display) 10 trước khi loop kiểm tra điều kiện là intCounter phải nhỏ hơn (<) 10 và mã nhảy ra khỏi loop sau đó.

For Loops

Ta dùng For loop khi biết trước sẽ lặp đi lặp lại việc thi hành nguồn mã bao nhiêu lần. Máy đếm trong trường hợp này (tự động tăng hay giảm tùy theo cách bố trí) sẽ thông báo chính loop của nó khi nào chấm dứt. Cú pháp như sau:

```
For intCounter = 1 to 10
  Response.Write(intCounter & "<br>")
Next
```

hoặc là:

```
For intCounter = 10 to 1 Step -1
  Response.Write(intCounter & "<br>")
Next
```

For loop còn có 1 dạng khác là **For each ... loop**, thường dùng để vận dụng các thành phần (hay yếu tố - elements) trong 1 bộ sưu tập (collection) tỷ như Array chẳng hạn:

```
'Bố trí 1 Array gồm các ngày trong tuần
Dim arrayWeekDays( ) As String = {"Mon", "Tue", "Wed", "Thu", "Fri"}
```

```
For each strDay in arrayWeekDays
  Response.Write(strDay & "<br>")
Next
```

Thay vì dùng số như trường hợp máy đếm intCounter trong 2 dạng For loop trên, ở đây **For each ... in** dùng 'strDay' để ấn định từng elements một theo thứ tự 'Mon', 'Tue', cho tới 'Fri' trong bộ sưu tập array mang tên 'arrayWeekDays'. Do đó, ta thấy browser sẽ hiển thị (display) 5 hàng chữ, mỗi hàng là ngày trong tuần.

Infinite Loop

Đề ý coi chừng trong khi dùng các loop mà ta phải tự quản lý việc tăng hay giảm máy đếm cho loop, nếu không khéo, ta sẽ rơi vào ... 'mê hồn trận' không có lối thoát gọi là **infinite loop**. Đây cũng là 1 trường hợp ngẫu nhiên mà lần đầu tiên bà con khám phá ra virus vì infinite loop nhanh chóng tiêu hao hay làm kiệt quệ các tài nguyên trong mạng và đôi khi có thể kéo cả mạng té bất ngờ (cause the site to crash).

Nếu ta muốn nhảy ra khỏi loop trước loop chấm dứt, ta có thể dùng keyword **Exit Do** (trong Do loop) hay **Exit For** (trong For loop), tỷ như:

```
'Bố trí counter loại Integer với giá trị 1
Dim intCounter As Integer = 1
```

```
'Bố trí Do loop và hiển thị (display) giá trị của counter ở browser
Do
  Response.Write(intCounter & "<br>")
  intCounter += 1
  If intCounter = 8 then
    Exit Do
```

End If

Loop While intCounter < 10

hay

```
For intCounter = 1 to 10
    Response.Write(intCounter & "<br>")
    If intCounter = 8 then
        Exit For
    End If
Next
```

Branching Logic

Branching logic cho phép nguồn mã thi hành ở những hướng khác nhau, tỷ như ta muốn nguồn mã tái thi hành ở một vị thế khác và có thể được dùng đi dùng lại nhiều lần trong các tình huống khác nhau.

Tổng quát có 2 loại branching logic:

- Functions
- Subroutines

Để phân biệt, ta để ý Functions tính toán các giá trị còn Subroutines thi hành các công việc (Functions compute values and Subroutines perform actions), như vậy Functions sẽ trả lại thông tin đã tính toán về nơi gọi Functions, ngược lại Subroutines thi hành công tác nào đó nhưng không trả lại gì hết (return nothing).

Functions

Cú pháp như sau:

```
Function FunctionName (Parameter1 As Type, ... , ParameterN As Type) As ReturnType
    'you code here
    Return ReturnValue
End Function
```

Vài VB.NET Function tiện dụng:

Sau đây là vài VB.NET Functions đã soạn sẵn trong .NET Framework ta có thể dùng trong trang Web của khóa:

Các functions về Date và Time

Function	Phân diễn tả công dụng
date(datetime)	Trả lại 1 số từ 1 tới 31 biểu thị ngày trong tháng.
dayofweek	Trả lại 1 số từ 0 (Sunday) tới 6 (Saturday) biểu thị ngày trong tuần.
dateDiff(dateinterval, date1,date2[, firstdayofweek[, firstdayofyear]])	Trả lại 1 số diễn tả khoảng thời gian cách biệt giữa 2 ngày: ngày 1 (date1) và ngày 2 (date2). Số này có thể ở dưới dạng yyyy (year), q (quarter), m (month - tháng), y (day of year), d (day), w (weekday), ww (week), h(hour), n (minute) hay s (second).
hour(time)	Trả lại 1 số từ 0 tới 23 biểu thị giờ trong ngày.
isdate(datetime)	Trả lại 1 số Boolean xác minh datetime có phải là 1 ngày hợp lệ không?
minute(time)	Trả lại 1 số từ 0 tới 59 biểu thị phút trong giờ.
month(datetime)	Trả lại 1 số từ 1 tới 12 biểu thị tháng trong datetime.
now()	Trả lại dưới dạng datetime ngày giờ hiện thời
second(time)	Trả lại 1 số từ 0 tới 59 biểu thị giây trong phút.
year(datetime)	Trả lại 1 số biểu thị cho năm (có giá trị từ 1 tới 9999)

Các Functions về Toán Học

Function	Phân diễn tả công dụng
abs(value)	Trả lại giá trị tuyệt đối (absolute) của value.

atan(value)	Trả lại số arctangent của value.
cos(value)	Trả lại số cosin của value.
exp(value)	Trả lại số exponent của value.
fix(value)	Trả lại phần integer của value, rounding up cho các số âm (negative numbers).
hex(value)	Trả lại số hexadecimal của value (base 10 to base 16).
int(value)	Trả lại phần integer của value, rounding down cho các số âm (negative numbers).
log(value)	Trả lại số nature logarithm của value.
oct(value)	Trả lại số octal của value (base 10 to base 8).
rnd	Trả lại số ngẫu nhiên.
round(value [, dec])	Trả lại số rounds to integer or with dec (decimal places) với của value.
sin(vlaue)	Trả lại số sin của value.
sqrt(value)	Trả lại số căn 2 (squareroot) của value.
tan(value)	Trả lại số tangent của value.

Các Functions về hành chữ (Strings)

Function	Phân diễn tả công dụng
instr(start,]string1, string2[, compare)	Trả lại số biểu thị vị trí 1 của string 2 trong string 1, ngoài ra trả lại số 0. Compare có thể dưới dạng 0 (=BinaryCompare) hay 1 (=TextCompare).
left(string, length)	Trả lại 1 string chứa 1 số chữ (dài = length) bắt đầu từ phía bên trái của hàng chữ.
len(string variable)	Trả lại 1 số biểu thị chiều dài của string hay tổng số bytes mà variable chứa đựng.
mid(string, start[, length])	Trả lại 1 string chứa trong 1 string khác, bắt đầu từ vị trí start kéo dài với giá trị của length.
replace(expression, find, replace[, start[, count[, compare]]])	Thay thế giá trị find trong biểu thức expression bằng giá trị replace bắt đầu từ vị trí start với số lần thay thế count. Lưu ý giá trị mặc định của count là -1 - nghĩa là thay thế tất cả gì truy tìm được, compare dùng dưới dạng 0 hay 1 như trong phần instr.
right(string, length)	Trả lại 1 string chứa 1 số chữ (dài = length) bắt đầu từ phía bên phải của hàng chữ.

Subroutines

Cú pháp như sau:

Sub SubroutineName (Parameter1 **As** Type, ... , ParameterN **As** Type)

'you code here

End Sub

Thí dụ subroutine Page_Load rất phổ thông mà ta thường dùng trong các trang ASP.NET :

Sub Page_Load (**Obj** **As** object, **e** **As** eventargs)

'you code here

End Sub

Các biến số (variables) trong () gọi là **thông số (parameters hay arguments)**, nếu hợp thành 1 nhóm, ta gọi **parameter list**. Subroutine chuyển và vận dụng các giá trị của thông số vào bên trong subroutine để thi hành 1 công tác nào đó.

Bài tập 1:

Mục đích:

Làm quen cách viết và sử dụng parameters cho Subroutine toán nhân với 2 số chẳng hạn.

1. Chạy ứng dụng Notepad và gõ hàng mã sau:

```
<%@Page Language="VB" %>
```

```
<script runat="server">
```

```
sub Page_Load(obj as object,e as eventargs)
```

```
    MultiplyNumbers(2,2)
```

```
    MultiplyNumbers(18,28)
```

```
    MultiplyNumbers(300,200)
```

```
end sub
```

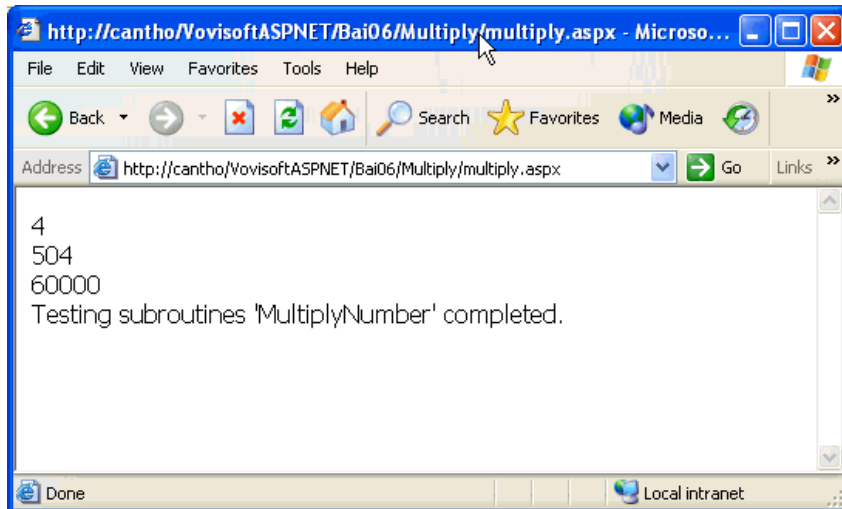
```
sub MultiplyNumbers(intA as integer,intB as integer)
```

```
    Response.Write(intA *intB &"<br>")
```

```
end sub
</script>

<html>
<body>
    Testing subroutines 'MultiplyNumber' completed.
</body>
</html>
```

2. Lưu trữ tập tin với tên multiply.aspx ở folder 'D:\Net\Vovisoft ASPNET\Bai06\Multiply' và chạy IE với URL như sau: <http://cantho/VovisoftASPNET/Bai06/Multiply/Multiply.aspx>



Phần ChúThích:

Ta nhận thấy khi gọi (call) 1 subroutine, ta chỉ cần dùng tên của subroutine kèm với các thông số (parameters) ta muốn gọi vào và vận dụng trong subroutine. Trong mã trên, ta gọi subroutine 'MultiplyNumber' 3 lần, trong đó ta sẽ vận dụng các thông số vào bài toán nhân và hiển thị (display) kết quả trong browsers với đối tượng (Object) Response và phương pháp (method) Write.

Lưu ý:

Trong VB.NET và ASP.NET, ta bắt buộc dùng parentheses () khi gọi các subroutines, khác với VBScript hay VB6 không cần dùng parentheses. Nếu ta gọi subroutine MultiplyNumber theo cú pháp sau, sẽ tạo lỗi:

MultiplyNumber 2, 2

Event Handler

Nếu để ý, ta thấy 1 trong những thông số (parameters) sử dụng ở bài tập trên là thông số về sự cố (EventArgs). Trong các trang Web của ASP.NET, sự cố có thể xảy ra bất cứ lúc nào, tỷ như user nhấp mũi chuột vào 1 nút nào đó trên Form hoặc nhấp vào 1 hình ảnh, do đó ta cần chuẩn bị và tạo ra cái gọi là **event handler** để có thể đáp ứng lại các sự cố đó. Cú pháp của event handler giống y chang như cú pháp của subroutine, sự khác biệt là nằm ở parameters list với thông số chỉ riêng cho loại **EventArgs**. Khi 1 sự cố khởi động, sự cố đó sẽ tạo ra các biến số nhằm diễn tả việc gì đã xảy ra và event handler sẽ dựa trên các biến số đó để phản ứng sao cho thích hợp.

Bài tập 2:

Mục đích:

Tạo 1 trang ASP.NET với nút bấm Submit và event **OnClick** liên hệ với Subroutine Button_Click để ... 'bắt quả tang' user vừa mới nhấp mũi chuột vào nút bấm đó.

1. Chạy ứng dụng Notepad và gõ hàng mã sau:

```
<%@ Page Language="VB" %>

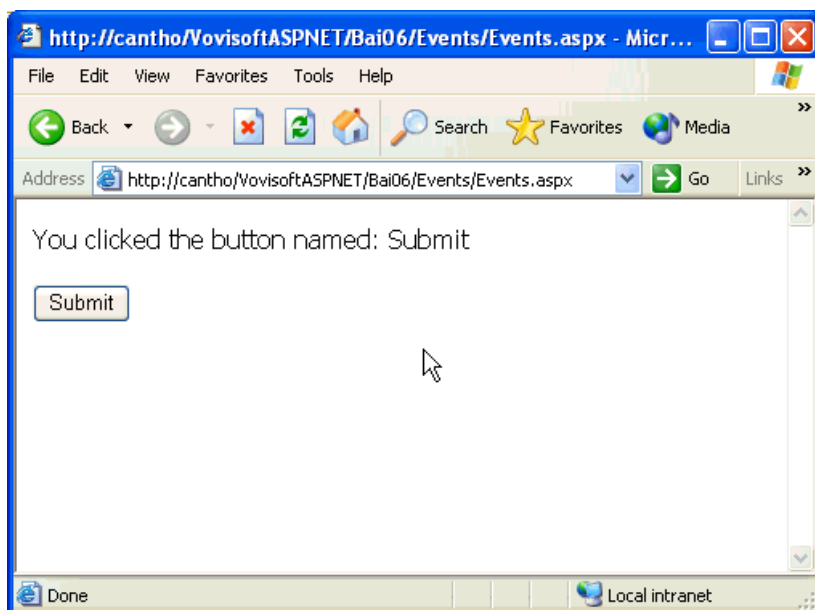
<script runat="server">
    Sub Button_Click(obj As Object, e As EventArgs)
        Response.Write("You clicked the button named: " & obj.Text)
    End Sub
</script>

<html>
<body>

    <form runat="server">
    <asp:button id="btSubmit" Text="Submit" runat=server
        OnClick="Button_Click"/><p>
    </form>

</body>
</html>
```

2. Lưu trữ tập tin với tên events.aspx ở folder 'D:\Net\Vovisoft ASPNET\Bai06\Events' và chạy IE với URL như sau: <http://cantho/VovisoftASPNET/Bai06/Events/Events.aspx>



Phần Chú Thích:

1. Thông số đầu tiên ở subroutine Button_Click là loại Object data type, obj As Object đại diện cho đối tượng (Object) gây ra sự cố, ở đây chính là nút bấm (button) mang tên 'btSubmit'.
2. Thông số thứ 2 là sự cố 'e' loại EventArgs, thông số này chứa mọi thông tin đặc trưng cho biến cố đang, sẽ hay đã xảy ra.
3. Trước khi user nhấp mũi chuột vào nút bấm này (lý lịch của nút bấm - ID là 'btSubmit') thì thông số 'e' trống rỗng, sau khi user nhấp vào nút bấm, 'e' chứa đủ thứ trong đó có sự cố 'Click'. Trong Form, ta đã sắp xếp nguồn mã để báo cho nút bấm 'btSubmit' hay rằng bất cứ khi nào sự cố Click khởi động, lập tức thì hành subroutine 'Button_Click', và như vậy, ta đã định nghĩa và xây dựng 1 event handler thành công.

Classes

Classes là phần ta xác định hay định nghĩa các đối tượng (Object) tỷ như để định nghĩa 1 cái đồng hồ, ta diễn tả kim giờ, kim phút, kim giây cùng các con số chỉ giờ, cách bố trí giờ giấc hay ngày tháng năm, ... Tương tự như thế, class định nghĩa đối tượng (Object) qua các đặc tính (properties) và các phương pháp (method) biểu thị đặc trưng cho class.

Ta nên nhớ rõ 1 điều: 'Mọi thứ trong .NET Framework hay VB.NET đều đại biểu cho classes'.

Cú pháp như sau:

```
Class classname
    properties
    subroutines
    functions
End Class
```

Trước khi dùng, nhớ instantiate class ra 1 đối tượng (Object).

Có hằng hà sa số **Base Classes** trong .NET Framework ta không thể nào kể ra xiết ở đây, tổng quát có thể gom vào những loại (categories) sau:

- String
- Collections và Arrays: tỷ như Arrays, Lists, Maps, Linked Lists, ...
- WinForms: dùng hiển thị (display) Windows và các Controls tỷ như Text Boxes, Combo Boxes, List Boxes, File Dialogs, ...
- Web Forms: phác thảo dùng cho mạng, ta sẽ đào sâu chi tiết ở bài kế.
- File Handling: dùng lướt qua lại (navigate) các file system trong máy hay trong mạng, kiểm tra đặc tính (properties) của files, read, modify hay write cũng như chuyển (move) và sao chép (copy) các tập tin hay folders.
- Registry Access: lướt qua lại, đọc hay viết nội dung của registry.
- Internet: nối vào mạng, tải lên hay tải xuống các tập tin.
- ADO.NET: nối vào các cơ sở dữ liệu (database) và vận dụng các records với 1 khái niệm mới về **disconnected data** cũng như sử dụng **XML** để chuyển data đi khắp mọi nơi mọi chỗ.

Inheritance

Inheritance nắm vai trò quan trọng trong classes và OOP (Object-Oriented Programming) vì tính chất kế thừa của nó. Ta không cần phải tạo ra 1 class mới hoàn toàn khi đã có 1 class tương tự như cái ta muốn (who wants to re-invent the wheel?) mà chỉ cần tạo ra nhánh (giống như nhánh cây) dựa vào gốc (base class). Trở lại trường hợp cái đồng hồ, tỷ như ta đã tạo ra 1 class gọi là Clock, rồi sau đó mới sực nhớ ra có 2 loại đồng hồ: analog và digital. Thay vì quẳng cái Clock class đi và tạo ra 2 cái mới Analog class và Digital class, ta có thể xây dựng class mới lấy Clock class làm cơ bản (base class) và thêm vào đó những gì đặc trưng đến hay liên hệ đến loại của Clock là Analog hay Digital. Tính chất kế thừa này đem lại vận hội mới cho các Kỹ Sư Tin Học khi xây dựng các ứng dụng vì làm cho việc phát triển, triển khai và bảo trì trở nên dễ dàng hơn, đỡ bị ... nhứt 'đầu' hay bạc ... 'râu'.

Cú pháp như sau với thí dụ về AnalogClock:

```
Class AnalogClock: Inherit Clock
    private ClockWound ASP.NET Boolean = False
    Sub WindClock( )
        ClockWound = True
    End Sub
End Class
```

Phương pháp (method) lập trình tổng quát

Ngoài việc nắm vững kỹ thuật và cú pháp lập trình, thiết tưởng, ta cũng nên bàn 1 cách tổng quát về phương pháp (method) xây dựng tiêu chuẩn cho việc lập trình các ứng dụng (nói chung) và ASP.NET (nói riêng). Nhớ đây chỉ là một số khái niệm cơ bản khi ta muốn phát triển các dự án ứng dụng lập trình VB.NET cho các trang ASP.NET trong khoá học này. Các tiến trình được nhắc đến ở đây nhằm mục đích hiểu rõ các bước cần thiết trong việc lập trình khi áp dụng cho một dự án thực tế trong công ty của mình.

1. Requirement Specifications (Đặc điểm kỹ thuật cần thiết)

Mặc dù khi trình bày các đề tài trong khoá này, ta chỉ sơ lược về các đặc điểm kỹ thuật cho các bài tập, bài làm ở nhà hay dự án nhưng ta có thể dựa vào đó mà linh động sắp xếp các đặc điểm kỹ thuật cần thiết phù hợp cho dự án của ta trong tương lai. Ngoài ra, ta có thể thêm bớt sao cho dự án của mình được thêm nhiều công dụng thích hợp yêu cầu đòi hỏi của khách hàng. Trên thực tế, Software Consultant phải phỏng vấn nhiều người trong công ty để có một hình ảnh đầy đủ vấn đề về Requirement Specifications.

2. Design (Thiết kế)

Từ Requirement Specifications bước qua giai đoạn Thiết kế. Lúc này ta sẽ thấy có nhiều điểm không được nêu rõ trong Requirement Specifications và cần làm cho sáng tỏ (clarification) hơn. Công việc này do Systems Analyst thực hiện. Nếu dự án lớn thì có Software Architect thiết kế tổng quát trước khi chia ra các Team Leaders (thường thường là Systems Analysts). Trong giai đoạn này có khi phải làm Prototype (thử mô hình mẫu) để biết chắc kỹ thuật mình dùng có đủ khả năng và thích hợp với nhu cầu dự án không. Thiết kế là giai đoạn quan trọng nhất trong chu kỳ phát triển một nhu liệu. Chẳng những ta nghĩ cách xây dựng nhu liệu, mà còn kế hoạch chi tiết cách thử lúc nào, ở đâu trong code. Ta phải tưởng tượng mọi hoàn cảnh bất thường (unusual scenarios) nhưng có thể xảy ra để tìm giải pháp đối phó. Trong các công ty nhỏ hoặc trung (small business), Web Master sẽ làm mọi chuyện từ ... 'cây kim' cho tới ... 'phi thuyền' khi thiết kế các trang ASP.NET.

3. Coding hay Implementation (Thảo chương)

Đây là giai đoạn thảo chương và debug. Trong phần Debug thì có Unit Test (thử từng bộ phận) và Integration Test (thử chung). Mỗi khi có sửa đổi một chút thì phải thử lại nhiều thứ nên nếu có thể viết Test Script để tự động hóa công việc thử này (gọi là Regression Test) thì tiết kiệm rất nhiều thời giờ. Nếu nhu liệu phải phản ứng nhanh chóng (good response) trong khi chạy Real-Time thì phải thử nó trong hoàn cảnh phải giải quyết nhiều thỉnh cầu cùng một lúc (gọi là Stress Test).

4. Acceptance Test (Chạy Thử)

Khái niệm tổng quát về việc kiểm tra Functional and Acceptance Testing như sau:

Sự kiểm tra Functional and Acceptance Testing cũng đóng góp một phần rất quan trọng trong việc lập trình. Một cách tổng quát, trong nhiều kiểu kiểm tra thì sự kiểm tra về mặt chức năng (Functional Testing) giúp người triển khai kiểm soát lại xem phần lập trình ứng dụng đó có phù hợp với mọi chức năng đặt ra trước theo yêu cầu của dự án. Còn sự kiểm tra về mặt thừa nhận (Acceptance Testing) là để kiểm soát xem ứng dụng đó có phù hợp với hoàn cảnh hay môi trường sử dụng hay không, tỷ như chạy thử trong các nền Windows khác nhau, trong các phiên bản Browser khác nhau, trong các loại Browser khác nhau hay cùng phiên bản nhưng screen resolution khác nhau, ...

Trong nhiều trường hợp ở tại công ty của khách hàng, Software Consultant chứng kiến các giai đoạn chạy thử để xem nhu liệu xử lý mọi chuyện đúng như liệt kê trước đây. Các công chuyện xử lý chạy có nhanh đủ không, nhất là trong trường hợp nhu liệu phải giải quyết rất nhiều thỉnh cầu cùng một lúc. Trong những hoàn cảnh bất bình thường, nhu liệu có đứng vững không hay 'té' bất ngờ. Nếu dự án lớn, trước Acceptance Test còn có thêm một giai đoạn gọi là Factory Test khi Software Consultant đến tận công ty để chứng kiến ta chạy thử.

5. Commissioning, Roll-Out (Áp dụng)

Khi cho áp dụng rồi là bắt đầu giai đoạn Bảo đảm (Warranty) và Bảo trì (Maintenance). Bảo trì là thăm viếng lại nhu liệu để chữa trị Bug (fixing bugs) hay sửa đổi (modification) hay làm thêm (enhancement). Lúc bấy giờ ta sẽ thấy giá trị của một nhu liệu được chú thích tỉ mỉ. Thường thường nhu liệu ta bảo trì là do người khác viết hoặc do chính mình viết từ lâu rồi, không còn nhớ nữa, nên nếu có kèm documentation và được chú thích rõ ràng thì công việc bảo trì sẽ dễ dàng hơn rất nhiều.

Khi lập trình cho một công việc hay dự án nhỏ, có khi ta không chú ý nhiều đến giai đoạn Requirement Specifications. Sau này lúc đã viết code rồi mới khám phá ra việc mình làm không đúng như điều khách hàng muốn thì rất phiền. Nếu Specifications đã viết rõ ràng thì ta có thể xin thêm tiền thay đổi (variation), nhưng có khi khách hàng vẫn trách là ta không chuyên nghiệp (professional) và có thể mình mất khách trong tương lai.

Tiến trình 6 là trở lại tiến trình ban đầu nhằm mục đích kiểm tra Chu Kỳ Phát Triển Nhu Liệu nhiều lần để giảm thiểu lỗi và nhất là khi có sự thay đổi trong phần specifications hầu có thể nâng cấp 1 cách hiệu quả.

Tóm tắt

Ở bài này, ta đã bàn sơ lược về .NET Framework và tham khảo vài chi tiết về cú pháp của ngôn ngữ lập trình VB.NET cùng phương pháp (method) lập trình để giúp ta xây dựng và phát triển các trang Web 1 các hiệu quả.

Một cách tổng quát, ta có 5 loại biến số (variables) và 3 loại logic. Các logic Conditional, Looping và Branching này rất cần thiết trong việc giúp ta tái thi hành các nguồn mã một khi thoả mãn điều kiện đã định trước. Ta cũng phân biệt được sự khác nhau giữa Functions và Subroutines và lướt sơ qua khái niệm cùng cách vận dụng các event cũng như classes và khái niệm Inheritance trong OOP.

Trong bài kế 'Web Form - Part I', ta sẽ tham khảo về Web Form và vai trò của nó trong ASP.NET. Web Form là 1 khái niệm mới và hấp dẫn trong môi trường mạng vì Web Form cho phép ta kiểm soát ... 'động tĩnh' của user qua các objects hiển thị (display) trong User Interface nhưng lại nằm ở phía Server. Wow!

Download Source Code

[Nguồn mã bài tập 1](#)

[Nguồn mã bài tập 2](#)

Bài làm ở nhà

Câu hỏi 1: Is VB.NET case sensitive?

Câu hỏi 2: Trong VB6 ta có biến số variant, chuyện gì đã xảy ra cho variant ở VB.NET?

Câu Hỏi 3: Khi nào ta dùng For loop và khi nào dùng While loop?

Bài làm 1: Tạo 1 trang ASP.NET dùng class PERSON để biểu thị về người và object đại diện cho chính bạn với đặc tính (properties) giới tính (Nam/Nữ), màu tóc (xanh, đen, muối tiêu, bạc, ...), màu mắt (đen, đỏ, ...), màu da (ngâm, bánh mật, đen, ...) và ngày tháng năm sinh. Dùng đặc tính (properties) dayofweek của datetime để xác định ngày bạn sinh là ngày nào trong tuần và dùng 1 nút bấm Submit để gọi phương pháp (method) này cũn gnhư hiển thị (display) kết quả ở browser.