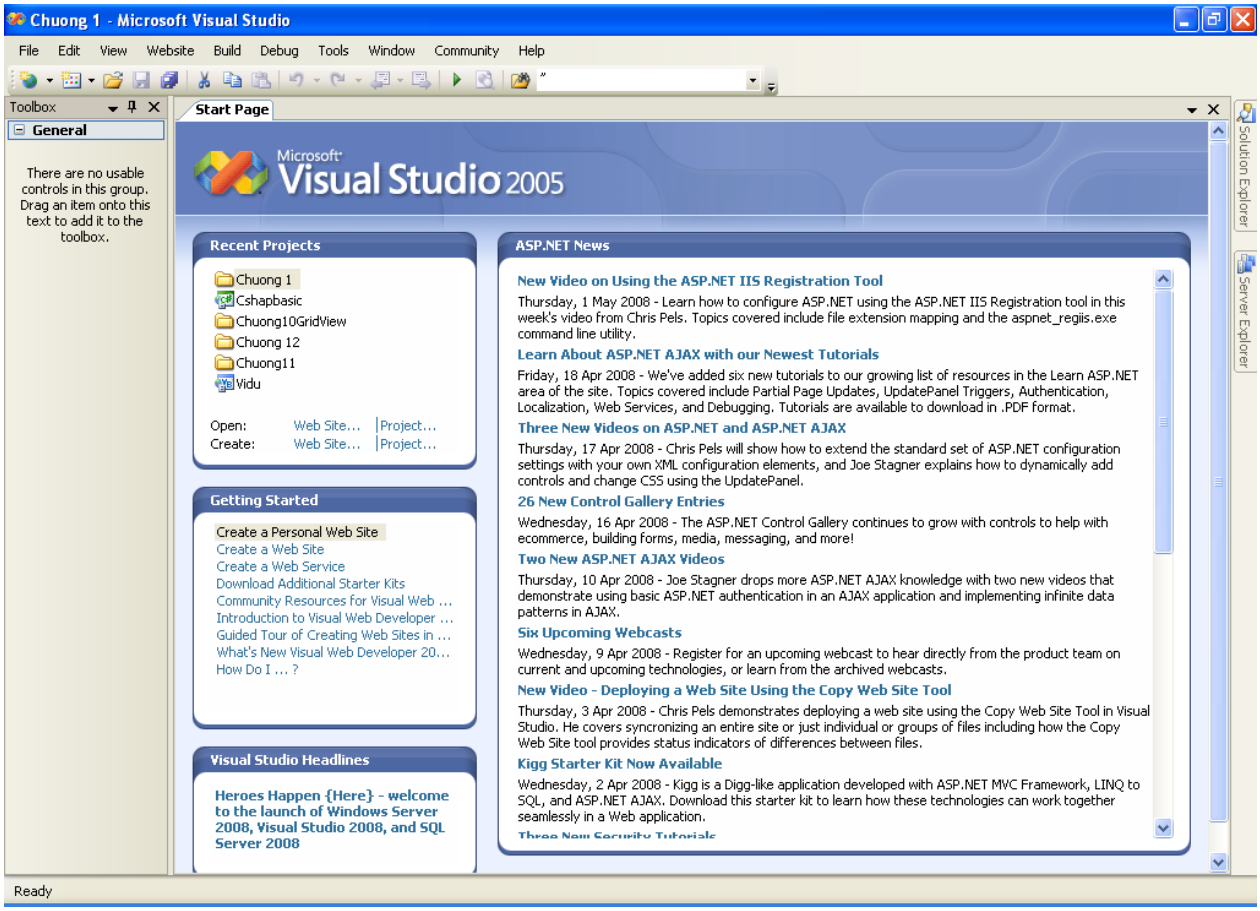
# Cấu trúc ASP.NET Framwork và cơ bản về C#

# Chương 1. Giới thiệu chung về cấu trúc ASP.NET Framwork và cơ bản về C#

## I. Giới thiệu chung về ASPNetFramwork

Trong giáo trình này chúng ta sẽ học ASP.NET trên IDE VisualStdio2005(Bạn có thể sử dụng Viusal Web Develop 2005 ).

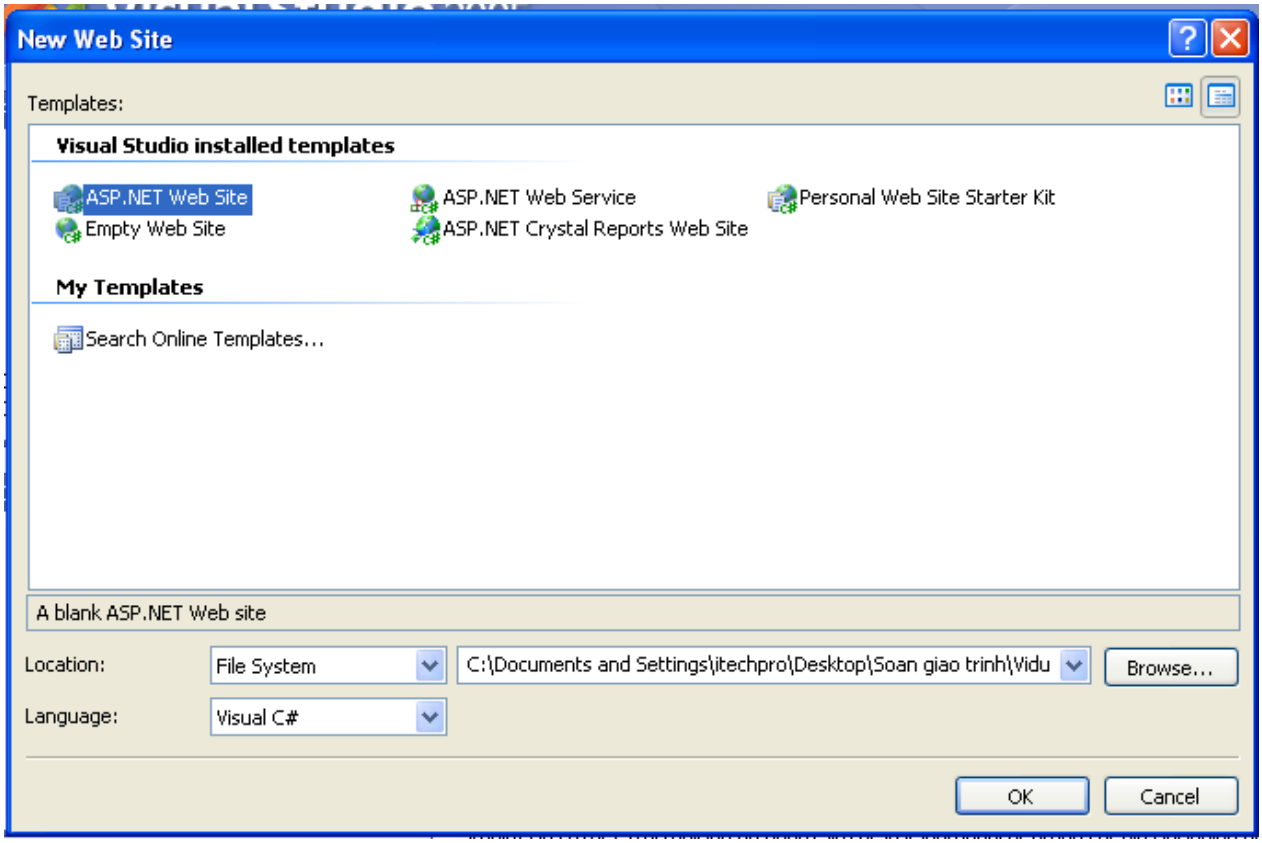
Để tạo một Wesite mới bạn khởi động VS. giao diện của nó sẽ hiện ra như sau:




Hình 1

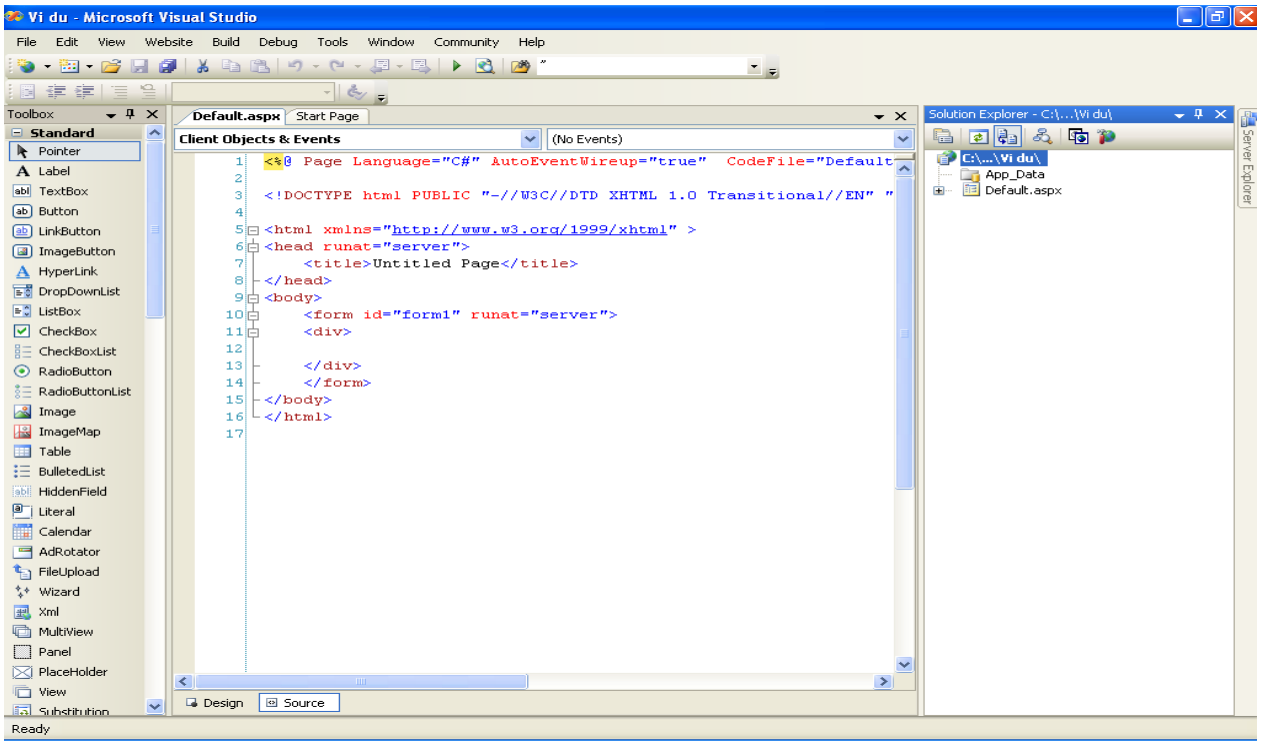
Trong Box Recent Project bạn chọn "Web site..." ở dòng Create

Hộp thoại New Website hiện ra bạn chọn ASP.NET WebSite rồi chọn thư mục bạn để Website và ngôn ngữ kịch bản để bản viết Web và nhấn OK




Hình 2

VS sẽ tạo ra một website với tên của Website là tên bạn vừa đặt và mặc định sẽ có một trang Default.aspx như hình sau:




Hình 3

## 1. ASP.NET và .NET FRAMEWORK

ASP.NET là một phần của .NET FrameWork Để xây dựng trang asp.NET bạn cần thêm vào các đặc tính của .netframework. NetFrameWork chứa đựng hai phần FrameWork Class Library và Commom Language Runtime.

### 1.1 Hiểu về Framework Class Library

Framework chứa đựng hàng nghìn lớp mà bạn có thể sử dụng trong ứng dụng của mình. Ví dụ một vài lớp của .Net Framework

-Lớp File: cho phép bạn tạo file, sửa, xoá hay kiểm tra sự tồn tại của file trên đĩa cứng…

-Lớp Graphics: Cho phép bạn làm việc với nhiều kiểu của ảnh, bạn cũng có thể tạo ra các ảnh từ các phương thức trên lớp này.

-Lớp SmtpClient: Cho phép bạn gửi thư.

Hiểu về Namespaces: hơn 13 nghìn lớp trong Netframework. Đây là một con số rất lớn, Microsoft đã chia các lớp cùng xử lý về một vấn đề gì đó vào các không gian tên chung hay namespaces.

Một Namespace đơn giản là một danh mục, ví dụ tất cả các lớp thao tác với File và thư mục chúng ta đưa vào một namespaces chung gọi là System.IO, hay tất cả các lớp làm việc với SqlServer có thể đưa vào namespace System.Data.SqlClient.

Các namespaces chung nhất trong net:

. System

. System.Collections

. System.Collections.Specialized

. System.Configuration

. System.Text

. System.Text.RegularExpressions

. System.Web

. System.Web.Caching

. System.Web.SessionState

. System.Web.Security

. System.Web.Profile

. System.Web.UI

. System.Web.UI.WebControls

. System.Web.UI.WebControls.WebParts

**1.2 Hiểu và Assembly:**

Một Assembly là một file dll trên đĩa cứng của bạn, nơi mà lưu trữ các lớp của .NET, ví dụ tất cả các lớp trong .ASP.NET Framework đều nằm trong Assembly System.web.dll.

Trước khi sử dụng các lớp trong dll bạn cần tạo một tham chiếu đến file dll này

**1.3 Hiểu về Commom Language Runtime(CLR)**

Phần thứ 2 của NetFramework là CLR chịu trách nhiệm về thực thi mã ứng dụng của bạn.

Khi bạn viết ứng dụng bằng ngôn ngữ C#, VB.NET hay bằng một ngôn ngữ bất kỳ trên nền NetFramwork mã của bạn sẽ được không bao giờ biên dịch trực tiếp thành mã máy. Thay vào đó chúng được biên dạng sang ngôn ngữ đặc tả MSIL (Microsoft intermediate Language).

MSIL nhìn rất giống với ngôn ngữ hướng đối tượng Assembly, nhưng không giống kiểu ngôn ngữ Assembly. MSIL là ngôn ngữ bậc thấp và phụ thuộc vào Platform.

Khi ứng dụng của bạn thực thi, mã MSIL  là "just in time" biên dịch sang mã máy bởi JITTER(just in time compiler)

Như vậy khi bạn viết các lớp trên .Net bằng bất kỳ ngôn ngữ nào khi bạn biên dịch sang Assembly bạn đều có thể sử dụng Assembly đó cho các ngôn ngữ khác.

**1.4 Hiểu về các điều khiển trên Asp.net**

Các điều khiển asp.net là phần quan trọng nhất trong ASP.NET Framework. một Control ASP.NET là một lớp mà thực thi trên server và đưa ra nội dung trên trình duyệt. ASP.NET có hơn 70 control mà bạn có thể sử dụng trong xây dựng ứng dụng web của bạn và cơ bản nó chia ra các nhóm control sau:

**Standard control**: bao gồm các điều khiển đưa ra các thành phần chuẩn của form như: Label, Button, TextBox…

**Validator Control**: là các control cho phép bản kiểm tra tính hợp lệ của các control cho phép nhập giá trị trên form.

**Rich Control**: là những điều khiển như FileUpload, Calendar…

**Data Control** là các điều khiển cho phép thao tác với dữ liệu

**Navigation Control**: là những điều khiển giúp bạn dễ dàng di chuyển giữa các trang trong website.

**Login control**: Là các điều khiển về bảo mật của ứng dụng cho phép bạn đưa ra các form đăng nhập, thay đổi mật khẩu…

**HTML Control**: cho phép bạn chuyển các điều khiển của HTML thành các điều khiển có thể làm việc trên server.

### 1.5 hiểu về điều khiển sự kiện trên server

phần lớn các điều khiển của asp.net hỗ trợ 1 hoặc nhiều sự kiện, ví dụ điều khiển ASP.NET Button hỗ trợ sự kiện Click, khi người sử dụng nhấn chuột vào Button một sự kiện sẽ được đưa ra và công việc này được xử lý trên server.

Ví dụ: Trang UnderstandEvent.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UnderstandEvent.aspx.cs" Inherits="UnderstandEvent" %>
<script runat="server">
    protected void Button1_Click(object sender, EventArgs e)
    {
        Label1.Text = TextBox1.Text;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Hiểu về sự kiện phía server</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <asp:Button ID="Button1" OnClick="Button1_Click"
            runat="server" Text="Button" />
        <hr />
        <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Trong ví dụ trên gồm 3 điều khiển của ASP.NET là TextBox, Label, và Button, mỗi khi người sử dụng nhập dữ liệu vào Textbox và nhấn vào Button sự kiện Button1_Click được đưa ra và điền dữ liệu từ TextBox và Label.

### 1.6 Hiểu về View State

Giao thức http là giao thức nền móng của WWW, là một giao thức chuẩn thực tế.mỗi lần bạn request một trang từ website, một dữ liệu mới được đưa ra, ASP.NET Framework có thể quản lý được vượt ra ngoài giới hạn của giao thức http, ví dụ bạn điền dữ liệu vào một điều khiển Label với thuộc tính Text của nó, dữ liệu này sẽ được lưu trữ qua nhiều trang web và chỉ thay đổi khi nó được gán lại giá trị.

Ví dụ sau sẽ đưa một một trang asp.net trong đó gồm 2 điều khiển Button và Label(Text của nó hiển thị số đếm), mỗi lần nhấn vào Button thì giá trị của Label tăng lên 1.

Ví dụ trang Understandstate.aspx:

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<script runat="server">
    protected void Button1_Click(object sender,EventArgs e)
    {
        Label1.Text = Convert.ToString(int.Parse(Label1.Text) + 1);
    }
</script>
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Under Stand State</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="Label1" runat="server" Text="0"></asp:Label>
        <asp:Button ID="Button1" OnClick="Button1_Click" runat="server"
Text="Button" />
    </div>
    </form>
</body>
</html>
```

Nếu bạn mở View Source code của trang Understandstate.aspx trên bạn sẽ thấy như sau:

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="/wEPDwULLTE4OTg4OTc0MjUPZBYCAgQPZBYCAgEPDxYCHgRUZXh0BQEzZGRkz0aT
ZTJffZRUP11aiDXbPGQGitk=" />
<input type="hidden" name="__EVENTVALIDATION" id="__EVENTVALIDATION"
value="/wEWAgKK7LjKBwKM54rGBgzCWqbCIzq33uVXHF19FzfdGPsJ" />
```

Đây là 2 file hidden trong form nó chứa giá trị Text của Label khi trang được postback nó sẽ ghi nhớ text đó và sẽ khởi tạo lại giá trị của Label khi trang Load.

### 1.7 Hiểu về trang asp.net

Sử dụng Code-Behind

Thay vì sử dụng <script runat=”sever”/> </script> ngay trên trang asp.net. người ta đưa ra thêm một trang gọi là Code-behind chứa các mã lệnh thực hiện trên trang asp.net.

Ví dụ như phần đầu của chương khi ta tạo ra một website thì mặc định sẽ tạo ra một lớp Default.aspx và nó sẽ kèm theo một trang Default.aspx.cs

Trang Default.aspx

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

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

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

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

Trang Defautl.aspx.cs

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

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

    }
}
```

### 1.8 Điều khiển sự kiện của trang asp.net

Khi chạy trang asp.net thì vòng đời của trang asp.net gồm các sự kiện

1.PreInit

2. Init

3. InitComplete

4. PreLoad

5. Load

6. LoadComplete

7. PreRender

8. PreRenderComplete

9. SaveStateComplete

10. Unload

Sử dụng thuộc tính Page.IsPostBack

Với sự kiện Load của trang thì khi tải trang lên thì có một sự kiện nào đó được đưa ra, nếu có nghĩa mỗi lần load lại trang nó lại thực hiện công việc đó, còn nếu ta đưa thêm vào thuộc tính Page.IsPostBack thì ta có thể điều khiển được sự kiện nào được thực hiện và sự kiện nào không khi trang đươc tải lại.

# II Cơ bản về lập trình C# lập trình trong trang ASP.NET

## 1. Kiểu dữ liệu.

C# đưa ra các kiểu dữ liệu dựng sẵn rất tiện ích, phù hợp với một ngôn ngữ lập trình hiện đại. Bảng sau đây sẽ miêu tả một số kiểu dữ liệu chính trong C#

| Kiểu C# | Kiểu .Net | Số Byte | Mô tả |
|---|---|---|---|
| byte | Byte | 1 | số nguyên không dấu từ 0 đến 255 |
| char | Char | 2 | Kiểu ký tự Unicode |
| bool | Boolean | 1 | Giá trị true/false |
| sbyte | Sbyte | 1 | Số nguyên có dấu, từ -128 đến 127 |
| short | Int16 | 2 | Số nguyên có dấu từ -32768 đến 32767 |
| ushort | Int16 | 2 | Số nguyên không dấu từ 0 đến 65.535 |
| int | Int32 | 4 | Số nguyên có dấu –2.147.483.647 đến 2.147.483.647 |
| uint | Int32 | 4 | Số nguyên không dâu 0 đến 4.294.967.295 |
| float | Single | 4 | kiểu dấu chấm động, giá trị xấp xỉ từ 3,4E-38 đến 3,4E+38, với 7 chữ số có nghĩa. |
| Double | Double | 8 | Kiểu dấu chấm động có độ chính xác gấp đôi, giá trị xấp xỉ từ 1,7E-308 đến 1,7E+308, với 15,16 chữ số có nghĩa |
| Decimal | Decimal | 8 | Có độ chính xác đến 28 con số và giá trị thập phân, được dùng trong tính toán tài chính, kiểu này đòi hỏi phải có hậu tố m hoặc M kèm theo sau. |

## 2. khai báo biến

Cú pháp: Kiểu Tên_biến;

Ví dụ:

string giatri_chuoi;

int giatri_nguyen;

chú ý biến có thể bao gồm các chữ cái, chữ số(không được đứng đầu) và ký tự _ (nối)

biến trong C# phân biệt chữ hoa và chữ thường.

## 3. Sử dụng các trình bày

### a. trình bày if – if else

Khi bạn cần kiểm tra một điều kiện nào đó trước khi thực hiện công việc, hoặc kiểm tra điều kiện nếu đúng thì làm việc còn khác thì không làm bạn có thể dùng trình bày if – if else

cú pháp:

```
if(điều_kiên)
{
      //thực hiện công việc
}

if(điều_kiên)
{
      // thực hiện công việc 1
}
else
{
      //thực hiện công việc 2
}
```

Lưu ý bạn có thể dùng nhiều cặp if – else lồng nhau:

Ví dụ:

Vd1

```
      if (conn.State != ConnectionState.Open)
          conn.Open();
```

Vd2

```
      if (1 > 2)
          MessageBox.Show("1>2");
      else
          MessageBox.Show("2>1");
```

### b, Sử dụng trình bày switch case

Khi công việc có nhiều lựa chọn và tuỳ vào từng trường hợp để bạn đưa ra công việc phù hợp với điều kiện đưa vào bạn có thể dùng trình bày switch case.

Ví dụ:

```
string giatri = Request.QueryString["abc"];
switch (giatri)
{
    case "a":
        //thuc hien cong viec a
        break;
    case "b":
        //thuc hien cong viec b
        break;
    default:
        //thuc hien cong viec mac dinh
        break;
}
```

**c, Sử dụng trình bày for**

Ví dụ

```
string giatri;
    for (int i = 0; i < 10; i++)
        giatri += i.ToString();
    MessageBox.Show(giatri);
```

Khi làm việc với mảng hay trong trường hợp thực hiện một công việc trong khoảng nào đó chúng ta có thể dùng trình bày for.

**d, Sử dụng trình bày while**

thực hiện công  việc trong khi điều kiện đúng

Ví dụ

```
int i = 0;

while (i < 5)

{

   Console.WriteLine(i.ToString());

   i++;

}
```

**e, Sử dụng trình bày do while**

ngược lại với while – do while làm việc cho đến khi điều kiện đúng thì thoát.

Ví dụ

```
int i = 0;
do
{
    MessageBox.Show(i.ToString());
```

```
        i++;
    } while (i < 3);
```

**f, Sử dụng trình bày break (*để thoát khỏi vòng lặp*)**

Ví dụ

```
int i = 0;
do
{
    MessageBox.Show(i.ToString());
    i++;
    if (i == 1)
        break;
} while (i < 3);
```

**g, Sử dụng trình bày continue.**

Ví dụ

```
int j = 0;
for ( int i = 0; i < 5; i++ )
{
    j++;
    if ( j > 2 )
    {
        MessageBox.Show(j.ToString());
        continue;
    }
}
```

**h, Sử dụng trình bày return(*được sử dụng trong các hàm để trả về giá trị cụ thể cho hàm*)**

Ví dụ

```
public int sum(int a, int b)
{
    return a + b;
}
```

**k, Sử dụng trình bày goto.**

Ví dụ

```
int i = 0;
int j = 0;
while (i < 5)
{
    i++;
```

```
            j++;
            if (j == 2)
                goto jumpeddoutofloop;
        }
    jumpeddoutofloop:
        Console.WriteLine("I jumped out");
```

## 4. Trang asp.net

Trang asp.net có đuôi mở rộng là .aspx và kèm theo một lớp phục vụ ẩn đằng sau(Code behind).

Để viết code C# trong trang aspnet ta có thể khai báo và sử dụng trực tiếp trong trang asp.net, trong file code behind, hoặc từ một thành phần thư viện và ta gọi vào.

### 4.1 Viết code C# trong file .aspx:

về cơ bản bạn dùng các các thẻ sau

- <% %> bạn có thể khai báo biến hoặc viết các hàm, lớp trong thể này,
- <%= %> với thẻ này bạn dùng để gọi giá trị của biến hay của 1 hàm nào đó,
- <%# %> lấy giá trị dùng trang các đối tượng ràng buộc dữ liệu.

Đây là một ví dụ đơn giản

Trang basic.aspx

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


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


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

<head runat="server">

   <title>Basic</title>

</head>

<body>

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

```
    <div>
        <%
            string abc = "Hello World!";
        %>
        Biến abc của bạn vừa khai báo có giá trị <%=abc %>
    </div>
    </form>
</body>
</html>
```

### 4.2 Viết code trong trang code behind

Vì trang aspnet của chúng ta kế thừa từ trang aspx.cs lên trong trang .aspx chúng ta muốn gọi dữ liệu từ biến hay hàm trong file .aspx.cs chúng ta phải khai báo với bổ ngữ truy cập protected hoặc public.

Ví dụ sau:

Trang codebehind.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <h3>Gán giá trị:</h3>
        <asp:Label ID="lblhello" runat="server"
Text="Label"></asp:Label><br /><br />
        <h3>Lấy giá trị từ code behind</h3>
        <%=_hello %>
    </div>
    </form>
</body>
</html>
```

Trang codebehind.aspx.cs

```
using System;

public partial class codebehind : System.Web.UI.Page
{
    protected string _hello;
    protected void Page_Load(object sender, EventArgs e)
    {
        _hello = "Hello World";
        lblhello.Text = _hello;
    }
}
```

Trong ví dụ trên có sử dụng một điều khiển asp.net là Label các bạn sẽ được học trong chương sau, ở chương này bạn hiểu nó là một điều khiển để hiển thị dữ liệu.

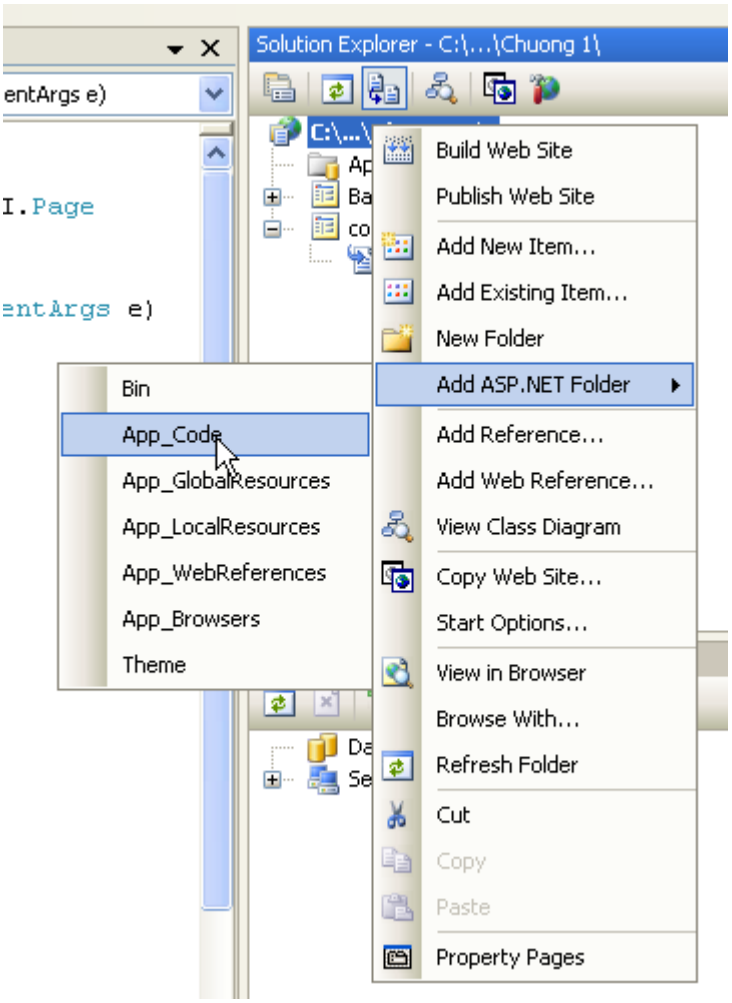
Bạn thấy trong phần code behind có khai báo một biến _hello kiểu string và bổ ngữ truy cập là protected trong sự kiện Page_Load(khi trang được tải lên) chúng ta gán _hello = "Hello World";  và sau đó gán giá trị cho Label bằng giá trị của _hello. Còn trong trang .aspx chúng ta có sử dùng thẻ <%= %> để lấy giá trị của _hello để in ra màn hình.

### 4.3 tạo một lớp thư viện

Để tạo một lớp thư viện phục vụ cho trang asp.net bạn có thể tạo một thành phần thư viện động DLL rồi nhập tham chiếu đến nó để sử dụng(chúng ta sẽ học nó trong phần asp.net nâng cao). Trong ứng dụng web ASP.NET Framework có một ASP.NET FOLDER là App_Code cho phép chúng ta viết các lớp thư viện ở đây và có thể sử dụng trong các trang của ứng dụng web.

để tạo thư mục App_code bạn làm theo các bước sau đây:

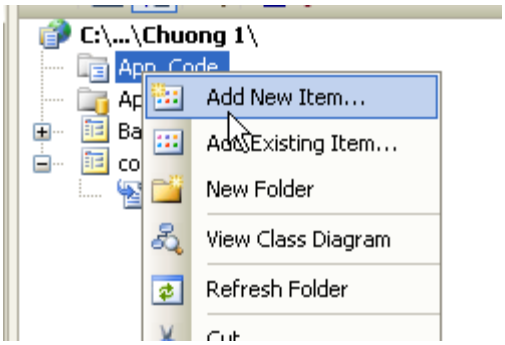
bước 1: nhấn chuột phải vào Solution và chọn theo đường dẫn của ảnh dưới đây.



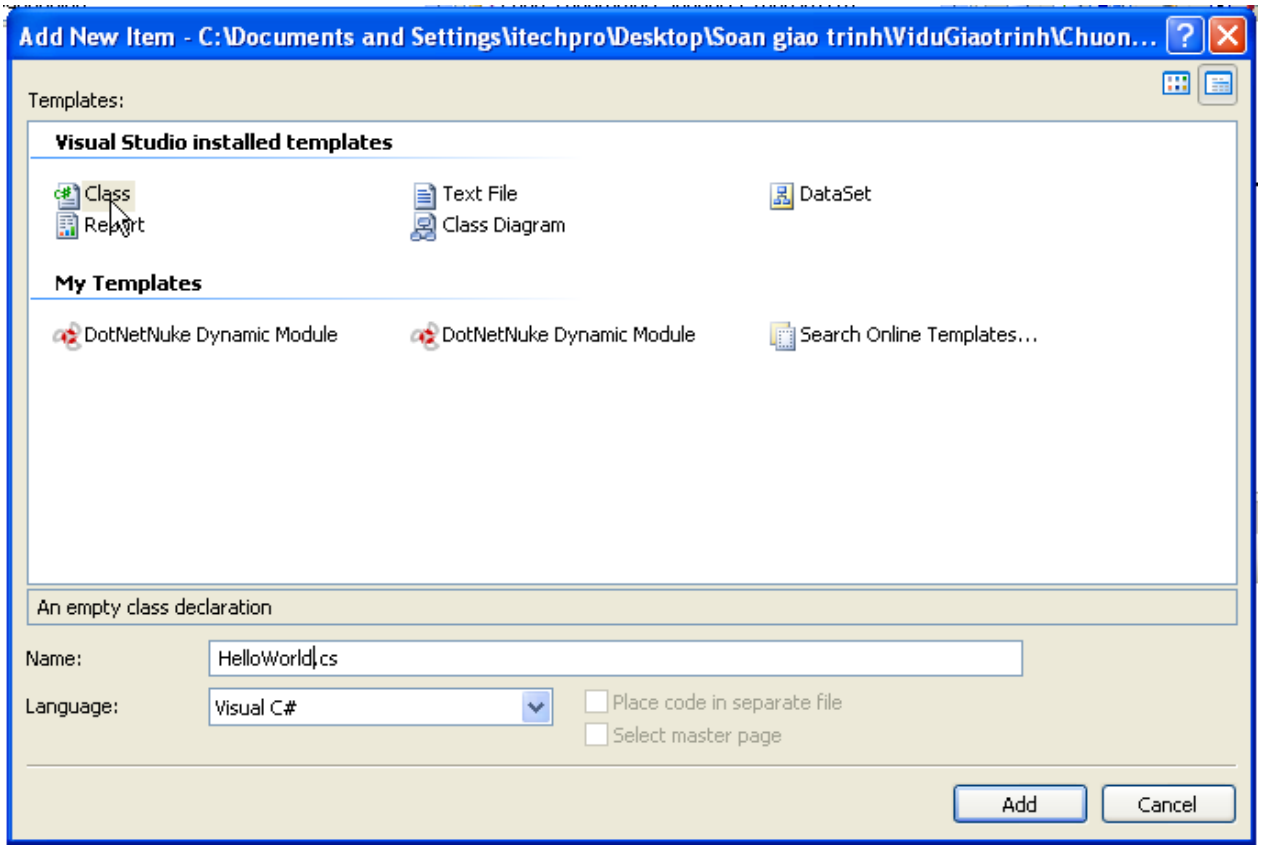

Trong ứng dụng web của chúng ta sẽ thêm vào một thư mục App_code


tại đây chúng ta có thể viết vào các lớp thư viện.

Để tạo một lớp thư viện trong thư mục này chúng ta nhấn chuột phải vào thư mục App_code rồi chọn Add New Item




Form Add New Item hiện ra




Bạn chọn Class và trong hộp TextBox Nam bạn nhập tên lớp muôn tạo và nhân nút Add.

***a, Định nghĩa lớp:***

Khai báo:

[Thuộc tính] [bổ sung truy cập] Class [Tên lớp] : [Lớp cơ sở]

```
{
  //các biến, phương thức hay thuộc tính của lớp
}
```

Ví dụ: Lớp HelloWorld.cs

```
class HelloWorld
  {
      public string SayMessage()
      {
          return "Hello World";
      }
  }
```

Trong ví dụ trên phương thức SayMessage sẻ về chuỗi "Hello World".

***b, Sử dụng định nghĩa truy cập***

Public: một lớp, một phương thức, hay thuôc tính khi sử dụng từ khoá này sẽ không bị hạn chế truy cập

Protected: Lớp, Phương thức, Thuộc tính chỉ được sử dụng ở lớp này hoặc lớp được dẫn xuất.

Internal: Một lớp, phương thức, thuộc tính Internal chỉ được truy cập trong một thành phần Assembly(file DLL).

Private: Một lớp Private, phương thức hoặc thuộc tính chỉ có thể truy cập tại chính lớp đó.

***c, Hàm và thủ tục***

Bạn có thể hiểu đơn giản hàm phải có giá trị trả về còn thủ tục như một đoạn mã chỉ thực hiện khi được chúng ta gọi. thủ tục còn được gọi là hàm không kiểu, hàm và thủ tục trong C# gọi chung là phương thức.

Ví dụ hàm:

```
public static int Sum(int _a, int _b)
 {
     return _a + _b;
 }
```

Trên là một hàm dùng để tính tổng của hai số, như bạn thấy trả về dữ liệu cho hàm chúng ta dùng từ khoá return, bổ sung truy cập public có ý nghĩa hàm được sử dụng trong toàn ứng dụng, từ khoá static đây là một phương thức tĩnh lên có thể sử dụng mà không cần phải khai báo khởi tạo đối tượng

Ví dụ về thủ tục

```
public static void HelloProcedure(string _bien)
{
    System.Web.HttpContext.Current.Response.Write(_bien);
}
```

Sử dụng lớp HellWorld trong trang aspx của chúng ta

Trang UseHelloworld.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UseHelloworld.aspx.cs" Inherits="UseHelloworld" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Sử dụng Lớp Hello World trong thư mục App_Code</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="lblHello" runat="server"
Text="Label"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Trang UseHelloworld.aspx.cs

```
using System;
public partial class UseHelloworld : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        lblHello.Text = HelloWorld.sayMessage();
    }
}
```

Vì phương thức sayMessage trong lớp HelloWorld là một phương thức tĩnh lên ta không cần khởi tạo lớp để sử dụng.

## 5. Cơ bản về lớp trong C#

### 5.1 Khai báo Field và thuộc tính

Ví dụ về Field

```
    public class HelloWorld
    {
        public string _Message;
        public string SayMessage()
        {
            return _Message;
        }
    }
```

Trong đoạn mã trên bạn thấy Field _Message được khai báo kiểu string và bổ ngữ truy cập là public, và _Message được trả về giá trị bởi phương thức SayMessage().

Ví dụ về thuộc tính

```
public class HelloWorld
```

```
{
    public string _Message;
    public string Message
    {
        get { return _Message; }
        set { _Message = value; }
    }
}
```

Một thuộc tính Message được khai báo ở trên gồm 2 phương thức get trả về giá trị cho Message và phương thức set thiết lập giá trị cho Message. Thuộc tính Message ở trên là phương thức vừa đọc vừa ghi. nếu bạn xây dựng thuộc tính chỉ đọc thì bạn chỉ cung cấp phương thức get hay thuộc tính chỉ ghi bạn cung cấp cho thuộc tính đó phương thức set.

### 5.2 Phương thức khởi dựng của lớp

Phương thức khởi dựng là phương thức đặc biệt của lớp, nó được gọi tự động khi khởi tạo mới lớp đó.bạn sử dụng phương thức khởi dụng để khởi tạo các private fields chứa đựng trong lớp. Phương thức khởi dựng của lớp phải trùng với tên của lớp, 1 phương thức của lớp có thể có đối số hoặc không có đối số, và có thể có nhiều phương thức khởi dựng cho lớp nhưng các đối số trong các phương thức phải khác nhau.

Ví dụ:

Xây dựng lớp: Construction.cs

```
using System;
public class Construction
{
    int _giatri1;
    int _giatri2;
      public Construction()
      {
        _giatri1 = 0;
        _giatri2 = 0;
      }

    public Construction(int _giatri1, int _giatri2)
    {
        this._giatri1 = _giatri1;
        this._giatri2 = _giatri2;
    }

    public int Sum()
    {
        return _giatri1 + _giatri2;
    }
}
```

Trong lớp này chúng ta xây dựng hai phương thức khởi dựng một phương thức không có đối số và một phương thức có đối số, và một hàm tính tổng của 2 giá trị nó được sử dụng trang trang asp.net như sau:

Trang UseConstruction.aspx

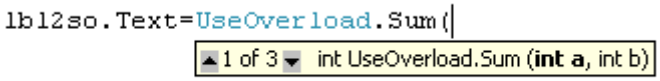
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UseConstruction.aspx.cs" Inherits="UseConstruction" %>

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Sử dụng phương thức khởi dựng của lớp</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="lblhello" runat="server"
Text="Label"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Trang UseConstruction.aspx.cs

```
using System;

public partial class UseConstruction : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
      Construction construc = new Construction(5, 6);
      lblhello.Text = "Giá trị là: " + construc.Sum().ToString();
   }
}
```

### 5.3 Overloading phương thức

Khi một phương thức được overloaded có nghĩa là hai phương thức có tên trùng nhau nhưng các đối số của nó phải khác nhau. Khi trong lớp của bạn có các phương thức overload thì bạn gọi hàm VS sẽ xuất hiện như sau để bạn có thể dễ dàng chọn được phương thức mình cần gọi.




Ví dụ:

Bạn tạo một lớp
Lớp UseOverload.cs

```
using System;

public class UseOverload
{
    public static int Sum(int a, int b)
```

```
    {
        return a + b;
    }

    public static int Sum(int a, int b, int c)
    {
        return a + b + c;
    }

    public static int Sum(int a, int b, int c, int d)
    {
        return a + b + c + d;
    }
}
```

Trong lớp này gồm 3 hàm tính tổng lần lượt được đưa vào 2,3,4 đối số

Sử dụng lớp này trong trang ASP.NET

Trang Overloading.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <h1>Chồng hoá phương thức tính tổng</h1>
     Tổng 2 số:<asp:Label ID="lbl2so" runat="server"
Text="Label"></asp:Label><br /><br />
     Tổng 3 số:<asp:Label ID="lbl3so" runat="server"
Text="Label"></asp:Label><br /><br />
     Tổng 4 số:<asp:Label ID="lbl4so" runat="server"
Text="Label"></asp:Label><br /><br />
    </div>
    </form>
</body>
</html>
```

Trang Overloading.aspx.cs

```
using System;

public partial class Overloading : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        lbl2so.Text = UseOverload.Sum(5, 5).ToString();
        lbl3so.Text = UseOverload.Sum(5, 5, 5).ToString();
        lbl4so.Text = UseOverload.Sum(5, 5, 5, 5).ToString();
    }
}
```

Trong lớp này bạn gọi lần lượt các phương thức tính tổng với 2,3,4 đối số để truyền giá trị vào các Label tương ứng cùng tên.

Kết xuất của chương trình:


### 5.4 Khai báo không gian tên (Namespaces)

Nếu bạn từng lập trình java chắc hẳn bạn đã quen với khái niệm packed mà bạn để đóng gói các lớp mà bạn xây dựng có đặc tính chung(miêu tả hay xử lý vấn đề gì đó). Trong .Net cũng vậy từ khoá Namespaces cũng có nhiệm vụ như packed trong java.

.Net cung cấp cho chúng ta các Namespaces như:

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

Và để sử dụng các Namespaces trong C# bạn cần sử dụng từ khoá using. Một Namespaces có thể chứa các Namespaces con, và trong Namespace con nhất chứa các lớp thành viên

Ví dụ

Bạn tạo ra hai lớp phép cộng và phép trừ để thực hiện các phép toán tương ứng như sau:

Lớp Phepcong.cs

```
using System;

namespace Vidu.Tinhtoan
{
    public class Phepcong
    {
        public static int Sum(int a, int b)
        {
            return a + b;
        }
    }

}
```

Và lớp Pheptru.cs

```
using System;

namespace Vidu.Tinhtoan
{
    public class Pheptru
    {
        public static int Minus(int a, int b)
        {
            return a - b;
        }
    }
}
```

Như bạn thấy hai lớp này nằm trong Namespaces Vidu.Tinhtoan, thì Vidu là Namespaces lớn nhất, còn Namespaces Tinhtoan là con của Vidu và trong tính toán chứa các lớp Phepcong và Pheptru.

Sử dụng Namespaces này trong trang asp.net

Trang Namespaces.aspx

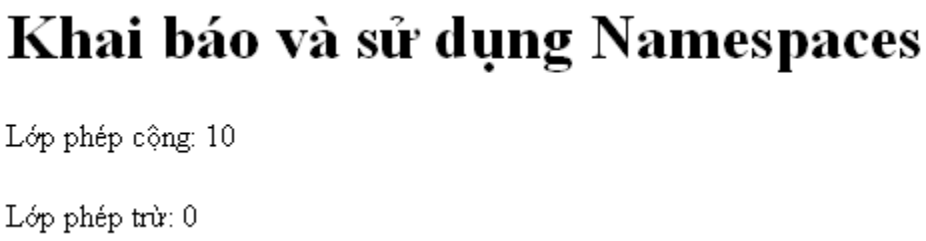
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Namespaces.aspx.cs" Inherits="Namespaces" %>

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
      <h1>Khai báo và sử dụng Namespaces</h1>
      Lớp phép cộng:
        <asp:Label ID="lblcong" runat="server"
Text="Label"></asp:Label><br /><br />
      Lớp phép trừ:
        <asp:Label ID="lbltru" runat="server" Text="Label"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Lớp Namespaces.aspx.cs

```
using System;
using Vidu.Tinhtoan;
public partial class Namespaces : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        lblcong.Text = Phepcong.Sum(5, 5).ToString();
        lbltru.Text = Pheptru.Minus(5, 5).ToString();
    }
}
```

Như ví dụ trên bạn thấy chúng ta sử dụng namespace Vidu.Tinhtoan giống với các Namespace khác mà Microsoft cung cấp cho chúng ta.

Kết xuất của chương trình




### 5.5 Lớp Partial

.Net cho phép chúng ta tạo ra một lớp trong nhiều file khác nhau mỗi File cung cấp hay xử lý một công việc gì đó trên lớp đó.

Ví dụ sau đây chúng ta sẽ tạo một lớp Calculator với 2 phương thức cộng và trừ nằm trên hai File khác nhau.

File Calminus.cs

```
using System;

namespace Vidu.Tinhtoan
{
    public partial class Calculator
    {
        public static int Minus(int a, int b)
        {
            return a - b;
        }
    }
}
```

File Calsum.cs

```
using System;

namespace Vidu.Tinhtoan
{
    public partial class Calculator
    {
        public static int Sum(int a, int b)
        {
            return a + b;
        }
    }
}
```

Như các bạn thấy hai file Calsum và Calminus chứa đựng cùng một tên lớp Calculator và trong mỗi File chứa đựng một phương thức riêng là thành phần của lớp đó.

Sử dụng lớp  này hoàn toàn giống với việc sử dụng một lớp khác.

File UsePartial.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Sử dụng lớp Partial</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
     <h1>Lớp Partial</h1>
     Kết quả cộng:
        <asp:Label ID="lblcong" runat="server"
Text="Label"></asp:Label><br /><br />
     Kết quả trừ:
        <asp:Label ID="lbltru" runat="server" Text="Label"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Và File UsePartial.aspx.cs

```
using System;
using Vidu.Tinhtoan;

public partial class UsePartial : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        lblcong.Text = Calculator.Sum(5, 5).ToString();
        lbltru.Text = Calculator.Minus(5, 5).ToString();
    }
}
```

### 5.6 Kế thừa và trừu tượng hoá một lớp

Khi một lớp được kế thừa từ một lớp thứ 2 thì nó được thừa hưởng tất cả các thuộc tính và phương thức không private từ lớp thứ nhất.

Kế thừa được sử dụng thông suốt trong .NetFrameWork, ví dụ trong tất cả các trang ASP.NET đều được kế thừa từ Lớp System.Web.UI.Page và tất cả các lớp trong .Net đều được dẫn xuất từ lớp cơ sở System.Object.

Ví dụ sau chúng ta sẽ đưa ra 2 lớp TelevisionProduct và ComputerProduct được kế thừa từ lớp BaseProduct.

Ví dụ:

```
using System;
public class BaseProduct
{
    decimal _price;
    public decimal Price
    {
```

```
        get { return _price; }
        set { _price = value; }
    }
}

public class ComputerProduct : BaseProduct
{
    string _processor;
    public string Processor
    {
        get { return _processor; }
        set { _processor = value; }
    }
}

public class TelevisionProduct : BaseProduct
{
    bool _isDHTV;
    public bool isDHTV
    {
        get { return _isDHTV; }
        set { _isDHTV = value; }
    }
}
```

Trong ví dụ trên bạn thấy hai lớp ComputerProduct và TelevisionProduct được kế thừa từ lớp BaseProduct, trong lớp BaseProduct có thuộc tính Price lên hai lớp kế thừa sẽ được kế thừa thuộc tính này.

Khi kế thừa từ một lớp khác, bạn có thể overload các thuộc tính và phương thức trong lớp này. Overloading một thuộc tính hay phương thức là một tiện ích khi bạn muốn thay đổi các ứng xử của phương thức hay thuộc tính đó trong lớp này.

Để Overload một phương thức hay thuộc tính từ lớp cơ sở, thì thuộc tính hay phương thức này phải được đánh dấu với từ khoá vitual hay abstract của C# hay trong VB.NET là Overridable hoặc MustOverride.

Ví dụ, chúng ta sẽ đưa ra hai lớp ProductBase và OnSaleProduct được kế thừa từ lớp ProductBase nhưng nó sẽ overload một thuộc tính từ lớp ProductBase.

Ví dụ:

```
using System;

public class ProductBase
{
    decimal _price;
    public virtual decimal Price
    {
        get { return _price; }
        set { _price = value; }
    }
}

public class OnSaleProduct : ProductBase
{
    public override decimal Price
    {
```

```
        get{ return base.Price / 2;}
        set{base.Price = value;}
    }
}
```

Trong ví dụ trên Lớp OnSaleProduct được kế thừa từ lớp ProductBase và Override thuộc tính Price, ở lớp này muốn chỉ ra giá bằng một nửa ở lớp cơ sở. chú ý với VB.NET thì từ khoá base là MyBase.

Bạn có thể dùng từ khoá abstract khi khai báo một lớp để như đánh dấu lớp này yêu cầu kế thừa. Bạn không thể tạo đối tượng một lớp trừu tượng, để sử dụng một lớp trừu tượng bạn phải dẫn xuất một lớp mới từ lớp trừu tượng và tạo đối tượng trong lớp dẫn xuất.

Ví dụ:

```
public abstract class BaseEmployee
{
   public abstract decimal Salary
   {
      get;
   }

   public string Company
   {
      get { return "Hoa Sen"; }
   }
}

public class SaleEmployee:BaseEmployee
{
   public override decimal Salary
   {
      get { return 66.666; }
   }
}
```

Trong ví dụ trên bạn thấy Lớp SaleEmployee được kế thừa từ lớp BaseEmployee và Overload thuộc tính Salary.

### 5.7 Khai báo Interface:

Một giao diện mà một danh sách các thuộc tính hay phương thức mà lớp kế thừa phải cài đặt. nếu một lớp cài đặt một giao diện, thì lớp này sẽ chứa tất cả các thuộc tính cũng như phương thức của giao diện này.

Ví dụ:

```
using System;

public interface IProduct
{
    decimal Price
    {
        get;
    }

    void SaveProduct();
}

public class MusicProduct : IProduct
```

```
{
    public decimal Price
    {
        get { return 20.99m; }
    }

    public void SaveProduct()
    {
        //Save Music Product
    }
}

public class BookProduct : IProduct
{
    public decimal Price
    {
        get { return 23.99m; }
    }

    public void SaveProduct()
    {
        //Save Book Product
    }
}
```

# Chương 2. Sử dụng các điều khiển Standard

Trong chương này các bạn sẽ học các điều khiển cở bản của ASP.NET Framework, đây là những điều khiển mà bạn thường xuyên sử dụng trong ứng dụng web của mình.

## I. Điều khiển hiển thị thông tin

### 1. Label

Một số thuộc tính

| Tên thuộc tinh | Ảnh hưởng |
|---|---|
| BackColor | Cho phép thay đổi màu nền của Label |
| BorderColor | Cho phép thiết lập đường viền của Label |
| BorderStyle | Hiển thị đường viền của Label theo các dạng Notset, None, Dotted, Dashed, Solid, Double, Groove, Ridge, Insert và OutSet. |
| cssClass | Cho phép đưa vào một lớp css |
| Font | Thiết lập thuộc tính của Font |
| ForeColor | Thiết đặt màu chữ nội dung của Label |
| Style | Cho phép gán các thuộc tính cho Label |
| ToolTip | Hiển thị nội dung khi di chuột vào điều khiển Label |
| ID | Tên của điều khiển |

Tuy điều khiển Label chứa đựng rất nhiều thuộc tính để kết xuất hiển thị nhưng với kinh nghiệm của mình trong quá trình thiết kế Web tôi khuyên bạn lên sử dụng thuộc tính cssClass để sử dụng một lớp css.

Ví dụ:

Gán thuộc tính trực tiếp

```
<asp:Label ID="Label1" BackColor="ActiveBorder"
BorderColor="ActiveCaption"
ForeColor="Blue" Font-Size="12pt" runat="server" Text="Diễn đàn
Đại Học Hoa Sen" />
```

Sử dụng cssClass

```
<asp:Label ID="Label2" CssClass="label"
runat="server" Text="Diễn đàn Đại Học Hoa Sen" />
```

Click phải vào tên ứng dụng trên cửa sổ Solution Explorer -> Add -> New Item -> Style Sheet

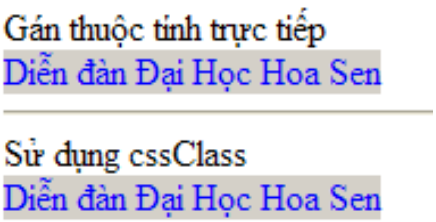
Copy đoạn mã sau:

```
.label

{ background-color:ActiveBorder; border-color:ActiveCaption; font-size:"12pt"; color:Blue; }
```

Thêm vào dưới

```
<title></title>

<style type="text/css">

@import url('Stylesheet1.css');

</style>
```

Kết quả




**2, Điều khiển Literal**

Giống với điều khiển Label bạn có thể dùng Literal để trình bày Text hoặc nội dung Html. Literal hỗ trợ một thuộc tính mà Label không hỗ trợ đó là: thuộc tính Mode gồm 3 giá trị là: Pass through, Endcode, transform.

Ví dụ

```
<asp:Literal ID="lit1" runat="server" Text="<hr/>" Mode="PassThrough" />

<asp:Literal ID="lit2" runat="server" Text="<hr/>" Mode="Encode" />

<asp:Literal ID="lit3" runat="server" Text="<hr/>" Mode="Transform" />
```

Hiên thị:

<hr/>

## II Điều khiển cho phép người dùng nhập liệu

### 1, Điều khiển TextBox

Bảng các thuộc tính:

| Thuốc tính | |
|---|---|
| Textmode | Kiểu hiển thị của Textbox gồm 3 giá trị: SingleLine-hiển thị 1 trường nhập liệu trên 1 dòng, MultiLine-hiển thị 1 trường nhập liệu nhiều dòng, Password-hiển thị 1 trường nhập mà Text sẽ được thay thế bằng các ký tự đặc biệt |
| AccessKey | Cho pheps bạn chỉ định phím để di chuyển tới control TextBox |
| AutoCompleteType | Cho phép bạn kết hợp với một lớp autoComplete với điều khiển TextBox. |
| AutoPostBack | Cho phép gửi dữ liệu lên server khi nội dung của TextBox thay đổi. |
| Columms | Cho phép chỉ định số cột được hiển thị trong TextBox |
| Enabled | Cho phép hay không cho phép nhập liệu trên TextBox |
| MaxLength | Cho phép quy đinh độ dài của dữ liệu mà một người sử dụng có thể nhập trên TextBox |
| ReadOnly | Cho phép chỉ đưa dữ liệu ra TextBox chứ không nhập dữ liệu vào. |
| Rows | Cho phép chỉ định số dòng hiển thị trong TextBox |
| TabIndex | Cho phép chỉ định thứ tự Tab của TextBox |
| Wrap | Cho phép chỉ định có word-wraps không khi thiết lập |

| | |
|---|---|
| | thuộc tính TextMode là Multiline |

Điều khiển TextBox hỗ trợ phương thức và sự kiện sau:

- Focus: cho phép thiết lập form khởi tạo ưu tiên tới TextBox

- TextChanged: Xảy ra trên Server khi nội dung TextBox thay đổi. để sự kiên này xảy ra bạn cần thiết đặt thuộc tính AutoPostback là true.

**2. Sử dụng điều khiển CheckBox**

Các thuộc tính

| | |
|---|---|
| AccessKey | Enables you to specify a key that navigates to the TextBox contro |
| AutoPostBack | nablesyou to post the form containing the CheckBox back to the<br>server automatically when the CheckBox is checked or unchecked |
| Checked | Cho phép bạn gán hoặc thiết đặt trạng thái chọn hay không chọn của CheckBox |
| Enabled | Cho phép hoặc không cho phép |
| TabIndex | Enables you to specify the tab order of the check box. |
| Text | Enablesyou to provide a label for the check box. |
| TextAlign | Enables you to align the label for the check box. Possible values are<br>Left and Right. |

CheckBox hỗ trợ phương thức va sự kiện

- Focus: Enables you to set the initial form focus to the check box.

- CheckedChanged: Raisedontheserverwhenthecheckboxischeckedorunchecked.

**3. Điều khiển RadioButton**

Điều khiển RadioButton luôn được sử dụng trong một nhóm và trong nhóm đó chỉ một RadioButton được chọn

Các thuộc tính

| | |
|---|---|
| Thuộc tinh | |

| | |
|---|---|
| AccessKey | Enablesyou to specify a key that navigates to the RadioButton control |
| AutoPostBack | Enables you to post the form containing the RadioButton back to<br>the server automatically when the radio button is checked or unchecked |
| Checked | Enables you to get or set whether the RadioButton control is checked. |
| Enabled | Enables you to disable the RadioButton |
| GroupName | Enables you to group RadioButton controls |
| TabIndex | Enables you to specify the tab order of the RadioButton control. |
| Text | Enables you to label the RadioButton control. |
| TextAlign | Enablesyou to align the RadioButton label. Possible values are Left<br>and Right. |

RadioButton hỗ trợ các phương thức và sự kiện

- Focus: Enables youto set the initial form focus to the RadionButton control.
- CheckedChanged: Raised on the server when the unchecked.

## III. Submitting Form Data

### 1. Điều khiển Button

Các thuộc tính:

| | |
|---|---|
| AccessKey | Cho phép chỉ định phím di chuyển tới điều khiển Button. |
| CommandArgument | Cho phép bạn chỉ rõ đối số được truyền tới lệnh thực hiện. |
| CommandName | Chỉ định tên một lệnh được truyền tới trong Command Event. |
| Enabled | Cho phép vô hiệu hoá điều khiển Button |
| OnClientClick | Cho phép chỉ định đến một hàm phía client khi nhấn vào Button. |

| | |
|---|---|
| PostBackUrl | Cho phép trả dữ liệu lên một trang khác. |
| TabIndex | Chỉ định thứ tự tab của Button. |
| Text | Nội dung text hiển thị trên điều khiển Button. |
| UseSubmitBehavior | Cho phép sử dụng javascript để trả dữ liệu lên một form. |

Các phương thức và sự kiện

- Focus: Cho phép thiết lập khi khơi tạo Form ưu tiên điều khiển TextBox.

- Click: Xảy ra khi điều khiển Button được nhấn.

- Command: Xảy ra khi điều khiển Button được nhấn. CommandName và CommandArgument được truyền qua sự kiện.

**2. Điều khiển LinkButton:** các phương thức và thuộc tính giống với điều khiển Button nhưng cách hiển thị của nó dưới dạng Text giống như thẻ <a> của HTML và có thể áp dụng thuộc tính css của thể <a> cho đối tượng LinkButton.

Học viên tự tìm hiểu thêm

**3. Điều khiển ImageButton**

Các thuộc tính và phương thức của điều khiển Button và thêm vào một số thuộc tính

| Thuộc tính | |
|---|---|
| ImageUrl | Chỉ đến đường dẫn của ảnh |
| ImageAlign | Cho phép căn chỉnh ảnh trong ImageButton các giá trị của nó có thể là: AbsBottom, AbsMiddle, Baseline, Bottom, Left, Middle, NotSet, Right, TextTop, and Top. |

**4. Sử dụng Client Scripts với điều khiển Button**

Cả ba điều khiển Button trên đều có thuộc tính OnClientClick, bạn có thể sử dụng thuộc tính này để thực hiện mã phía Client mà bạn cần khi điều khiển Button được nhấn.

Ví dụ.

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

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

<script runat=”server”>
      protected void btnDelete_Click(object sender, EventArgs e)
      {
            lblResult.Text = “All pages deleted!”;
      }
</script>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>HiepGia</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Button ID="Button1" OnClick="Button1_Click"
OnClientClick="return confirm('Bạn có chắc chắn?');" runat="server"
Text="Button" /><br />
        <asp:Label ID="Label1" runat="server" Text="HiepGia" />
    </div>
    </form>
</body>
</html>
```

Giải thích ví dụ: trên đoạn mã trên trong form gồm 2 điều khiển Button và Label. Trong điều khiển Button ta gán vào 2 sự kiện: OnClientClick phía Clien, và OnClick phía Server

OnClientClick sẽ thực hiện khi người dùng phía Client nhấn vào Button, trong mã Client trình bày một hàm confirm trong javascript với mục đính hỏi người dùng xác nhận việc thực hiện nào đó “Bạn có chắc chắn?” nếu người dùng chọn OK thì Sự kiện thứ 2 OnClick sẽ được thực hiện còn nếu chọn Cancel thì sự kiện phía Server không được thực hiện.

**5. Thực hiện chuyển trang**

Mặc định khi bạn nhấn vào Button nó sẽ thực hiện công việc ngay trên trang và trang của chúng ta sẽ Load lại một lần nhưng bạn có thể sử dụng thuộc tính PostBackUrl để chuyển sang một trang khác.

Ví dụ bạn có một trang ButtonSearch.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>HiepGia</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
     <asp:Label ID="lblSearch" runat="server" Text="Search:" />
     <asp:TextBox ID="txtSearch" runat="server" />
```

```
     <asp:Button ID="btnSearch" Text="GO!" runat="server"
PostBackUrl="ButtonSearchResult.aspx" />
    </div>
    </form>
</body>
</html>
```

Trong thuộc tính PostBackUrl của điều khiển btnSearch sẽ chuyển sang trang ButtonSearchResult

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

<script runat="server">
 void Page_Load(object sender, EventArgs e)
    {
        if (PreviousPage != null)
        {
            TextBox txtSearch =
(TextBox)PreviousPage.FindControl("txtSearch");
            lblSearch.Text = txtSearch.Text;
        }
    }
</script>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>HiepGia</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
     <asp:Label ID="lblSearch" runat="server" />
    </div>
    </form>
</body>
</html>
```

Trong đoạn mã trên ta để ý phương thức FindControl của PreviousPage nó sẽ tìm đến Điều khiển trên trang ButtonSearch.aspx mà ta cung cấp ID của nó.

### 6. Chỉ định một Button mặc định.

Ví dụ trên trang của bạn có nhiều điều khiển Button, khi nhập dữ liệu bạn lại quen nhập xong dữ liệu và nhấn nút Enter trên bàn phím, bạn có thể để mặc định nút cập nhật dữ liệu làm mặc định khi nhấn phím Emter. Như ví dụ dưới đây.

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

<script runat="server">
    void btnXacnhan_Click(object sender,EventArgs e)
    {
        lblThongbao.Text = txtHoten.Text;
    }
</script>
```

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>HiepGia</title>
</head>
<body>
    <form id="form1" runat="server" defaultbutton="btnXacnhan">
    <div>
        <asp:Label ID="lblHoten" runat="server" Text="Nhập họ tên" />
        <asp:TextBox ID="txtHoten" runat="server" />
        <hr />
        <asp:Button ID="btnXacnhan" OnClick="btnXacnhan_Click"
runat="server" Text="Xác nhận" />
        <asp:Button ID="btnBoqua" runat="server" Text="Bỏ qua" />
        <hr />
        <asp:Label ID="lblThongbao" runat="server" Text="" />
    </div>
    </form>
</body>
</html>
```

7. Điều khiển Command Event

Sự khác nhau giữa Command Event và Command Click là trong Command Event bạn có thể cung cấp Command Name và Command argument

Trong các ví dụ trước tôi đã lấy ví dụ về sự kiện Command Click nên bây giờ tôi sẽ lấy một ví dụ về Command Event để bạn so sánh.

Ví dụ trên trang bạn có 3 điều khiển Button như ví dụ dưới đây:

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

<script runat="server">
    void hcubiuChon(object sender, CommandEventArgs e)
    {
        if (e.CommandName == "language")
        {
            switch (e.CommandArgument.ToString())
            {
                case "C#":
                    lblComandEvent.Text = "CShap";
                    break;
                case "VBNET":
                    lblComandEvent.Text = "VB.NET";
                    break;
                case "JAVA":
                    lblComandEvent.Text = "Java";
                    break;
            }
        }
    }
</script>
```

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>HiepGia</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        Bạn chọn ngôn ngữ:
        <asp:Label ID="lblComandEvent" runat="server"
Text=""></asp:Label>
        <br />
        <asp:Button ID="btna" OnCommand="hcubiuChon"
CommandName="language" CommandArgument="C#" runat="server" Text="C#" />
        <asp:Button ID="btnb" OnCommand="hcubiuChon"
CommandName="language" CommandArgument="VBNET" runat="server"
Text="VB.NET" />
        <asp:Button ID="btnc" OnCommand="hcubiuChon"
CommandName="language" CommandArgument="JAVA" runat="server"
Text="Java" />
    </div>
    </form>
</body>
</html>
```

Cả ba điều khiển Button trên đều chứa Tên lệnh và đối số lệnh và tuỳ thuộc vào đối số lệnh khác nhau để ta có thể thực hiện một công việc tương ứng.

## IV. Điều khiển hiển thị ảnh

ASPNET bao gồm 2 điều khiển hiển thị ảnh. điều khiển Image và ImageMap.

### 1. Điều khiển Image.

Điều khiển này dùng để hiển thị ảnh giống với thẻ <Img> trong HTML

Các thuộc tính quan tâm

| Thuộc tính | |
|---|---|
| AlternateText | Nội dung thay thế khi lỗi đường dẫn của ảnh |
| DescriptionUrl | Cho phép bạn cung cấp một đường dẫn đến trang miêu tả chi tiết nội dung của ảnh |
| ImageAlign | Cho phép căn chỉnh ảnh lên quan tới các thành phần HTML khác trong trang và nó có thể là các giá trị sau:AbsBottom, AbsMiddle, Baseline, Bottom, Left, Middle, NotSet, Right, TextTop, and Top. |
| ImageUrl | Đường dẫn của ảnh trên điều khiển |

Ví dụ

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

<script runat="server">
    void Page_Load(object sender, EventArgs e)
    {
        Random rnd = new Random();
        switch (rnd.Next(3))
        {
            case 0:
                Image1.ImageUrl = "Images/images1.jpg";
                Image1.AlternateText = "Picture 1";
                break;
            case 1:
                Image1.ImageUrl = "Images/images2.jpg";
                Image1.AlternateText = "Picture 2";
                break;
            case 2:
                Image1.ImageUrl = "Images/images3.jpg";
                Image1.AlternateText = "Picture 3";
                break;
        }
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>HiepGia</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Image ID="Image1" runat="server" />
    </div>
    </form>
</body>
</html>
```

**2. Điều khiển ImageMap**

Cho phép bạn tạo bản đồ ảnh trên 1 ảnh hiển thị trên trang web mà người dùng phía Client có thể chọn vào một vùng của ảnh để thực hiện một công việc nào đó.

Ví dụ

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

<script runat="server">
    void ImageMap1_Click(object sender, ImageMapEventArgs e)
    {
        switch (e.PostBackValue)
        {
            case "top":
                lblResult.Text = "Day la phan dau";
```

```
                break;
            case "middle":
                lblResult.Text = "day la phan giua";
                break;
            case "under":
                lblResult.Text = "day la phan cuoi";
                break;
        }
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>HiepGia</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:ImageMap ID="ImageMap1" HotSpotMode="PostBack"
OnClick="ImageMap1_Click" ImageUrl="~/Images/jiwoo18034.jpg"
runat="server">
         <asp:RectangleHotSpot
            PostBackValue="top"
            Left="0"
            Top="0"
            Right="300"
            Bottom="150"
            AlternateText="Top" />
         <asp:RectangleHotSpot
            PostBackValue="middle"
            Left="0"
            Top="150"
            Right="300"
            Bottom="300"
            AlternateText="middle" />
         <asp:RectangleHotSpot
            PostBackValue="under"
            Left="0"
            Top="300"
            Right="300"
            Bottom="450"
            AlternateText="Under" />
        </asp:ImageMap>
        <asp:Label ID="lblResult" runat="server" />
    </div>
    </form>
</body>
</html>
```

Các thuộc tính của ImageMap giống với Image và thêm vào hai thuộc tính đó là:

- HotSpots: cho phép bạn điền tập hợp thông tin của HotSpots được chứa đựng trong điều khiển ImageMap.

- HotSpotMode:Cho phép bạn chỉ định  Enables you to specifythe behavior of the image map when you click a region.  Possible values are Inactive, Navigate, NotSet, and PostBack.

## V. Điều khiển Panel

Khi bạn cần đưa các điều khiển trang vào một nhóm để giải thích nghĩa cho nhóm đó hoặc có thể là để ẩn hoặc hiện nhóm điều khi nhấn 1 sự kiện nào đó trên trang của bạn, bạn có thể dùng điều khiển panel.

Một số thuộc tính của điều khiển panel mà bạn cần lưu ý là:

- DefaultButton: Cho phép bạn định nghĩa một button mặc định trong panel mà button mặc định này sẽ được thực hiện khi bạn nhấn phím Enter

- Direction: Cho phép bạn gán hoặc thiết đặt hướng hiển thị nội dung được đưa ra trong panel, có thể là các giá trị:NotSet, LeftToRight, and RightToLeft.

- GroupingText: Cho phép bạn trình bày Panel như 1 Fieldset với một chú giải riêng biệt

- HorizontalAlign: Cho phép bạn chỉ ra hướng ngang thể hiện nội dung của panel và nó có thẻ là các giá trị: Center, Justify, Left, NotSet, and Right.

- ScrollBars: Cho phép bạn hiển thị scrollbars khi bạn cố định chiều cao hoặc chiều rộng của panel và nội dung trong panel vượt quá độ rộng hoặc độ cao đó, nó có thể là các giá trị: Auto, Both, Horizontal, None, and Vertical.

Ví dụ

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

<script runat="server">
    void Page_Load(object sender, EventArgs e)
    {
        for (int i = 1; i < 100; i++)
        {
            buletnghenghiep.Items.Add("Nghề "+i.ToString());
        }
    }

    void hcubiuSothich(object sender, EventArgs e)
    {
        if (chkhtsothich.Checked == true)
            panelsothich.Visible = true;
        else
            panelsothich.Visible = false;
    }

    void hcubiuNghenghiep(object sender, EventArgs e)
    {
        if (chkhtnghenghiep.Checked == true)
```

```
            panelnghenghiep.Visible = true;
        else
            panelnghenghiep.Visible = false;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>HiepGia</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
     <asp:Panel ID="panelpage" runat="server" GroupingText="HiepGia">
      <table>
       <tr>
        <td>
         <asp:Panel ScrollBars="auto" ID="panelsothich" Width="200px"
Height="200px" runat="server" GroupingText="Sở thích">
             <asp:CheckBox ID="CheckBox1" Text="Bóng đá" runat="server"
/><br />
             <asp:CheckBox ID="CheckBox2" Text="Xem phim"
runat="server" /><br />
             <asp:CheckBox ID="CheckBox3" Text="Kinh doanh"
runat="server" />
         </asp:Panel>
        </td>
        <td>
         <asp:Panel ID="panelnghenghiep" Width="200px" Height="200px"
ScrollBars="Vertical" runat="server" GroupingText="Nghề nghiệp">
          <asp:BulletedList ID="buletnghenghiep" runat="server">
          </asp:BulletedList>
         </asp:Panel>
        </td>
       </tr>
       <tr>
        <td>
             <asp:CheckBox Checked="true" ID="chkhtsothich"
AutoPostBack="true" OnCheckedChanged="hcubiuSothich" Text="Hiển thị sở
thích" runat="server" /><br />
             <asp:CheckBox Checked="true" ID="chkhtnghenghiep"
AutoPostBack="true" OnCheckedChanged="hcubiuNghenghiep" Text="Hiển thị
nghề nghiệp" runat="server" /><br />
        </td>
       </tr>
      </table>
     </asp:Panel>
    </div>
    </form>
</body>
</html>
```

## VI. Điều khiển HyperLink

Điều khiển HyperLink cho phép tạo 1 link tới trang web khác, không giống với LinkButton, HyperLink không đệ trình dữ liệu lên server.

# Chương 3 Sử dụng các điều khiển Validation

Ở Chương trước chúng ta đã học về những điều khiển chuẩn của NetFrameWork3.5, chúng ta có thể dùng những điều khiển đó để thao tác với Webserver phía Server-Side, có thể để cập nhật dữ liệu. Nhưng có một tình huống đặt ra là các điều khiển đó có đảm bảo cho chũng ta cập nhật dữ liệu đúng và không xảy ra lôi?. Ví dụ trên Form của ta có trường nhập số điện thoại nhưng người sử dụng lại nhập vào là 1 dòng text như vậy sẽ gây ra lỗi nhập liệu. Với phiên bản trước của ASP.Net là asp thì để khắc phục lỗi đó chúng ta phải thực hiện viết mã JavaScript để bắt lỗi việc đó, còn với ASPNET nó đã cung cấp cho ta những điều khiển kiểm tra tính hợp lệ của các điều khiển nhập liệu trên Form. Trong chương này các bạn sẽ học về những điều khiển đó và tiếp theo là sẽ học cách mở rộng những  điều khiển đó theo ý muốn của chúng ta ví dụ bạn sẽ tạo một AjaxValidator để kiểm tra nhập liệu phía Client.

6 điều khiển Validation trong netframework3.5:

- RequiredFieldValidator: Yêu cầu người sử dụng nhập giá trị vào trường chỉ định trên Form
- RangeValidator: Kiểm tra giá trị nhập vào có nằm trong một khoảng nhỏ nhất và lớn nhất định trước hay không.
- CompareValidator: So sánh giá trị nhập có bằng một giá trị của trường khác trên Form hay không.
- RegularExpressionValidator: So sánh giá trị nhập với 1 biểu thức quy tắc nào đấy có thể hòm thư, điện thoại…
- CustomValidator: Bạn có thể tuỳ chỉnh đối tượng Validator theo ý của mình
- ValidationSummary: cho phép hiển thị tổng hợp tất cả các lỗi trên 1 trang.

## I. RequiredFieldValidator

1. **ý nghĩa:** với điều khiển này bạn có thể yêu cầu người dùng phải nhập giá trị vào 1 trường chỉ định trên Form.

### 2. Cách sử dụng:

Đưa điều khiển RequiredFieldValidator từ ToolBox(trong phần Validation) vào trong Form và thêm vào cho nó 2 thuộc tính

ControlToValidate: chỉ đến điều khiển sẽ được kiểm tra

Text(hoặc ErrorMessage): Thông báo lỗi khi kiểm tra

### 3. Ví dụ

Code 1: Trang RequiredValidator.aspx

```
<%@ Page Language="C#" %>
<script runat="server">
    void btnAccept_Click(object sender, EventArgs e)
    {
        if (Page.IsValid)
        {
            this.lblResult.Text = txtHoten.Text;
            this.txtHoten.Text = "";
        }
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Validator</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
       <asp:Label ID="lblHoten" runat="server" Text="Nhập vào họ tên"
/>
       <asp:TextBox ID="txtHoten" runat="server"></asp:TextBox>
       <asp:RequiredFieldValidator ID="RequiredFieldtxtHoten"
ControlToValidate="txtHoten" runat="server" Text="* Bạn phải nhập họ
tên"></asp:RequiredFieldValidator><br />
       <asp:Button ID="btnAccept" OnClick="btnAccept_Click"
runat="server" Text="Accept" /><br />
       <asp:Label ID="lblResult" runat="server" Text="" />
    </div>
    </form>
</body>
</html>
```

## II. Điều khiển CompareValidator

### 1. Ý nghĩa

Bạn có thể sử dụng CompareValidator để Kiểm tra giá trị nhập vào có nằm trong một khoảng nhỏ nhất và lớn nhất định trước hay không.

### 2. Cách sử dụng

Bạn đưa điều khiển CompareValidator từ hộp ToolBox vào Form và thiết lập cho nó một số thuộc tính sau:

- ControlToValidate: chỉ đến điều khiển cần kiểm tra
- Text(ErrorMessage): Nội dung thông báo lỗi
- MinimumValue: Giá trị nhỏ nhất thiết lập cho đối tượng
- MaximumValue: Giá trị lớn nhất thiết lập cho đối tượng

- Type: Kiểu so sánh, Có thể là các giá trị Interger,String, Double, Date và Currency.

### 3. Ví dụ

Code 2: Trang CompareValidator.aspx

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

<script runat="server">
     void btnAccept_Click(object sender,EventArgs e)
     {
         if (Page.IsValid)
         {
             this.lblThongbao.Text = txtDiem.Text;
         }
     }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>RangeValidator</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
     <table>
      <tr>
       <td>Vào điểm</td>
       <td><asp:TextBox ID="txtDiem" runat="server"></asp:TextBox></td>
       <td>
           <asp:RequiredFieldValidator ID="RequiredFieldDiem"
ControlToValidate="txtDiem" runat="server" ErrorMessage="Bạn phải nhập
điểm"></asp:RequiredFieldValidator>
          <asp:RangeValidator ID="RangeDiem" runat="server"
ControlToValidate="txtDiem" Type="Integer" MinimumValue="0"
MaximumValue="10" ErrorMessage="Điểm phải nằm trong khoảng từ 0 đến
10"></asp:RangeValidator>
       </td>
      </tr>
      <tr>
       <td colspan="3">
           <asp:Button ID="btnAccept" OnClick="btnAccept_Click"
runat="server" Text="Thực hiện" />
       </td>
      </tr>
      <tr>
       <td colspan="3">
        <hr />
        <asp:Label ID="lblThongbao" runat="server" />
       </td>
      </tr>
     </table>
    </div>
```

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

Trong ví dụ trên ta dùng hai đối tượng Validator cùng kiểm tra giá trị nhập trên điều khiển txtDiem, điều khiển RequiredFieldDiem dùng đề kiểm tra và yêu cầu nhập giá trị cho txtDiem còn điều khiển RangeDiem yêu cầu nhập giá trị trong txtDiem phải nằm trong khoảng từ 0 đến 10.

## III. Điều khiển RegularExpressionValidator

### 1. ý nghĩa

Điều khiển RegularExpressionValidator cho phép bạn so sánh giá trị nhập tại 1 trường nào đó trên Form với một quy tắc định trước. bạn có thể sử dụng các biểu thức quy tắc để đưa ra các chuỗi mẫu như là email addresses, Social Security numbers, phone numbers, dates, currency, amounts, and product codes.

### 2. Cách sử dụng

Bạn đưa điều khiển RegularExpressValidator vào Form của mình và thiết lập cho nó một số thuộc tính sau:

- ID: tên của điều khiển
- ControlToValidate: trỏ đến điều khiển cần kiểm tra
- Text(ErrorMessage): nội dung thông báo khi có lỗi
- ValidatorExpression: quy định mẫu nhập liệu như là hòm thư, số điện thoại…

### 3. Ví dụ

Sau đây sẽ là một ví dụ về việc yêu cầu người sử dụng phải cập nhật đúng địa chỉ của hòm thư.

Code 3: trang RegularExpressionValidator.aspx

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

<script runat="server">
    void btnAccept_Click(object sender, EventArgs e)
    {
        if (Page.IsValid)
        {
            lblThongbao.Text = txtEmail.Text;
        }
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>RegularExpressionValidator</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        Email:<asp:TextBox ID="txtEmail" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ControlToValidate="txtEmail"
ID="RequiredFieldValidator1" runat="server" ErrorMessage="Bạn phải nhập
địa chỉ hòm thư"></asp:RequiredFieldValidator>
        <asp:RegularExpressionValidator
ID="RegularExpressionValidator1" runat="server"
            ErrorMessage="Bạn nhập không đúng định dạng hòm thư"
            ControlToValidate="txtEmail"
            ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-
.]\w+)*"></asp:RegularExpressionValidator>
        <br />
        <asp:Button ID="btnAccept" OnClick="btnAccept_Click"
runat="server" Text="Accept" />
        <asp:Label ID="lblThongbao" runat="server"></asp:Label>
    </div>
    </form>
</body>
</html>
```

## IV. Điều khiển CompareValidator

### 1. ý nghĩa

Điều khiển CompareValidator có 3 kiểu khác nhau để kiểm tra giá trị nhập:

- Sử dụng để kiểm tra kiểu dữ liệu
- Sử dụng để so sánh giá trị nhập với một giá trị cố định
- Sử dụng để so sánh gia trị nhập với giá trị của một điều khiển khác trên Form

### 2. Cách sử dụng

Bạn đưa điều khiển CompareValidator vào Form và thiết lập cho nó một số thuộc tính sau:

- ControlToValidate: điều khiển của Form sẽ được kiểm tra
- ControlToCompare: Điều khiển dùng để so sánh giá trị
- Text(ErrorMessage): hiển thị nội dung thông báo lỗi kho có lỗi
- Type: Kiểu của giá trị sẽ được so sánh

- Operator: Toán tử so sánh. Có thể là các giá trị: DataTypeCheck, Equal, GreaterThan, GreaterThanEqual, LessThan, LessThanEqual, NotEqual.

### 3. Ví dụ

Ví dụ 4 sau sẽ yêu cầu nhập vào ngày sinh, nếu người sử dụng nhập vào không đúng dữ liệu dạng ngày thì sẽ có lỗi thông báo.

Code 4: Trang CompareValidator.aspx

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

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

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

        <asp:Label ID="Label1" runat="server" Text="Ngày
sinh"></asp:Label>
        <asp:TextBox ID="txtNgaysinh" runat="server"
Width="154px"></asp:TextBox>
        <asp:CompareValidator ID="CompareValidator1" runat="server"
            ControlToValidate="txtNgaysinh"
            ErrorMessage="Sai dữ liệu phải là kiểu ngày tháng"
Operator="DataTypeCheck"
            Type="Date"></asp:CompareValidator>
        <br />
        <asp:Button ID="Button1" runat="server" Text="Accept" />

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

Ví dụ sau đây sẽ hướng đưa ra trường hợp với Form tạo tài khoản trên một trên Web yêu cầu người đăng ký phải nhập mật khẩu 2 lần.

Code 5 trang

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

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

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

```
<head runat="server">
    <title>CompareValidator</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        <asp:Label ID="Label2" runat="server" Text="Nhập mật khẩu
"></asp:Label>
        <asp:TextBox ID="txtPass" runat="server"
TextMode="Password"></asp:TextBox>
        <br />
        <asp:Label ID="Label3" runat="server" Text="Nhập lại mật
khẩu"></asp:Label>
        <asp:TextBox ID="txtrePass" runat="server"
TextMode="Password"></asp:TextBox>
        <asp:CompareValidator ID="CompareValidator2" runat="server"
            ControlToCompare="txtPass" ControlToValidate="txtrePass"
            ErrorMessage="Nhập mật khẩu 2 lần phải giống
nhau"></asp:CompareValidator>
        <br />
        <asp:Button ID="Button2" runat="server" Text="Accept" />


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

## V. Điều khiển CustomValidator

### 1. ý nghĩa

Nếu những điều khiển Validator trên chưa đủ với bạn hoặc bạn muốn tạo một Validator riêng theo ý mình, bạn có thể dụng điều khiển CustomValidator, bạn có thể kết hợp CustomValidator với một hàm.

### 2. Cách sử dụng và Ví dụ

CustomValidator có 3 thuộc tính hay sử dụng là:

- ControlToValidator: điều khiển của Form sẽ được kiểm tra
- Text(ErrorMessage): hiển thị nội dung thông báo lỗi kho có lỗi
- ClientValidationFunction: tên của một hàm client-side để thực hiện kiểm tra trên client-side

CustomValidator hỗ trợ 1 sự kiện

ServerValidate: Sự kiện được đưa ra khi CustomValidator thực hiện kiểm chứng.

Ví dụ sau sẽ sử dụng sự kiện ServerValidate để kiểm tra độ dài của chuỗi được nhập trong điều khiển TextBox, nếu người nhập, nhập vào chuỗi có độ dài lớn hơn 20 ký tự thì điều khiển CustomValidator sẽ đưa ra thông báo lỗi.

Ví dụ:

Code 6 trang CustomValidator.aspx

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

<script runat="server">
    void CustomValidator1_ServerValidate(object source,
ServerValidateEventArgs e)
    {
        if (e.Value.Length > 20)
            e.IsValid = false;
        else
            e.IsValid = true;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

        <asp:Label ID="Label1" runat="server" Text="Ghi
chú"></asp:Label>
        <br />
        <asp:TextBox ID="TextBox1" runat="server" Height="95px"
TextMode="MultiLine"
            Width="218px"></asp:TextBox>
        <asp:CustomValidator ID="CustomValidator1" runat="server"
            ErrorMessage="Độ dài ghi chú phải nhỏ hơn 20 ký tự"
            ControlToValidate="TextBox1"
OnServerValidate="CustomValidator1_ServerValidate"></asp:CustomValidato
r>
        <br />
        <asp:Button ID="Button1" runat="server" Text="Accept" />

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

Ở ví dụ 6 trong hàm "CustomValidator1_ServerValidate" Tham số thứ 2 được truyền tới sự kiện ServerValidator để xử lý. Trong thực thể của lớp ServerValidateEventArgs có hai thuộc tính

- Value: Giá trị của trường trên Form sẽ được kiểm chứng.
- IsValid: Diễn tả việc kiểm chứng cho kết quả thành công hoặc sai.

Trong ví dụ tiếp theo tôi sẽ đưa ra cách sử dụng hàm kiểm chứng Client-side kết hợp với CustomValidator như thế nào, Trang này chỉ kiểm tra độ dài của chuỗi nhập vào bên trong TextBox, nhưng nó sẽ kiểm tra trên cả Server và Client.

Code 7.

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

<script runat="server">
    void CustomValidator1_ServerValidate(object source,
ServerValidateEventArgs e)
    {
        if (e.Value.Length > 20)
            e.IsValid = false;
        else
            e.IsValid = true;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>CustomValidator</title>
    <script language="javascript" type="text/javascript">
     function valComments_ClientValidate(source, args)
     {
        if (args.Value.length > 20)
            args.IsValid = false;
        else
            args.IsValid = true;
     }
    </script>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        <asp:Label ID="Label1" runat="server" Text="Ghi
chú"></asp:Label>
        <br />
        <asp:TextBox ID="TextBox1" runat="server" Height="95px"
TextMode="MultiLine"
            Width="218px"></asp:TextBox>
        <asp:CustomValidator ID="CustomValidator1" runat="server"
            ErrorMessage="Độ dài ghi chú phải nhỏ hơn 20 ký tự"
ClientValidationFunction="valComments_ClientValidate"
            ControlToValidate="TextBox1"
OnServerValidate="CustomValidator1_ServerValidate"></asp:CustomValidato
r>
        <br />
        <asp:Button ID="Button1" runat="server" Text="Accept" />

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

```
</html>
```

## VI. Điều khiển ValidationSummary

### 1. ý nghĩa

ValidationSummary cho phép bạn liệt kê tất cả các các lỗi kiểm tra trên trang từ những điều khiển validator vào một vị trí. Điều khiển này đặc biệt tiện ích với Form có độ rộng lớn.

### 2. cách sử dụng

Bạn đưa điều khiển ValidationSummary vào Form và thiết lập cho nó một số thuộc tính sau:

- DisplayMode: Cho phép bạn chỉ rõ định dạng hiển thị lỗi, nó có thể là các giá trị như BulletList, List, và SingleParagraph.
- HeaderText: Cho phép bạn hiển thị tiêu đề tóm tắt cho các lỗi.
- ShowMessageBox: Cho hiện thị một popup thông báo
- ShowSummary: Cho phép bạn ẩn ValidationSummary trên trang.

### 3. ví dụ

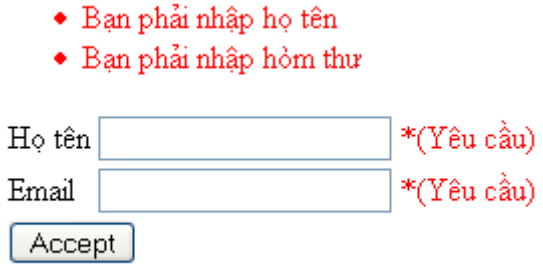
Code 8 Trang ValidationSummary.aspx

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

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

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

</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:ValidationSummary ID="ValSummary" runat="server" />
        <table>
            <tr>
                <td>
                    <asp:Label ID="Label1" runat="server" Text="Họ
tên"></asp:Label>
                </td>
                <td>
                    <asp:TextBox ID="txtHoten"
runat="server"></asp:TextBox>
                    <asp:RequiredFieldValidator
ID="RequiredFieldValidator1" runat="server"
```

```
                        ErrorMessage="Bạn phải nhập họ tên"
ControlToValidate="txtHoten">*(Yêu cầu)</asp:RequiredFieldValidator>
                </td>
            </tr>
            <tr>
                <td>
                    <asp:Label ID="Label2" runat="server"
Text="Email"></asp:Label>
                </td>
                <td>
                    <asp:TextBox ID="txtEmail"
runat="server"></asp:TextBox>
                    <asp:RequiredFieldValidator
ID="RequiredFieldValidator2" runat="server"
                        ErrorMessage="Bạn phải nhập hòm thư"
ControlToValidate="txtEmail">*(Yêu cầu)</asp:RequiredFieldValidator>
                    <asp:RegularExpressionValidator
ID="RegularExpressionValidator1" runat="server"
                        ErrorMessage="Hòm thư bạn nhập không đúng định
dạng"

ControlToValidate="txtEmail"></asp:RegularExpressionValidator>
                </td>
            </tr>
            <tr>
                <td colspan="2">
                    <asp:Button ID="Button1" runat="server"
Text="Accept" />
                </td>
            </tr>
        </table>

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

Code 8

Chạy ví dụ trên nếu ta không nhập dữ liệu cho các trường nó sẽ thông báo lỗi như sau:



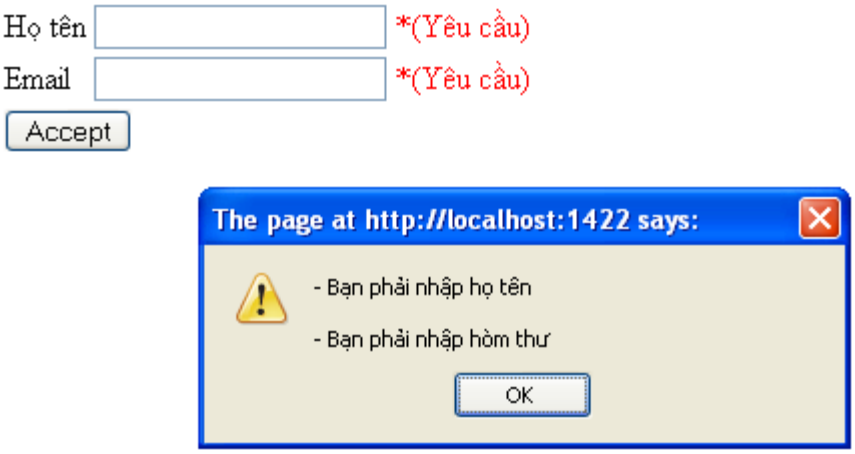

Cũng với ví dụ trên nếu trên điều khiển ValSummary ta thiết lập thuộc tính ShowMessageBox bằng True và ShowSummary với giá trị bằng False thì kết xuất của ví dụ thay vì hiển thị là một danh sách lỗi trên Form thì nó hiển thị một popup thông báo những lỗi trên trang.

Code 9

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

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

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

</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:ValidationSummary ShowMessageBox="true"
ShowSummary="false" ID="ValSummary" runat="server" />
        <table>
            <tr>
                <td>
                    <asp:Label ID="Label1" runat="server" Text="Họ
tên"></asp:Label>
                </td>
                <td>
                    <asp:TextBox ID="txtHoten"
runat="server"></asp:TextBox>
                    <asp:RequiredFieldValidator
ID="RequiredFieldValidator1" runat="server"
                        ErrorMessage="Bạn phải nhập họ tên"
ControlToValidate="txtHoten">*(Yêu cầu)</asp:RequiredFieldValidator>
                </td>
            </tr>
            <tr>
                <td>
                    <asp:Label ID="Label2" runat="server"
Text="Email"></asp:Label>
                </td>
                <td>
                    <asp:TextBox ID="txtEmail"
runat="server"></asp:TextBox>
                    <asp:RequiredFieldValidator
ID="RequiredFieldValidator2" runat="server"
                        ErrorMessage="Bạn phải nhập hòm thư"
ControlToValidate="txtEmail">*(Yêu cầu)</asp:RequiredFieldValidator>
                    <asp:RegularExpressionValidator
ID="RegularExpressionValidator1" runat="server"
                        ErrorMessage="Hòm thư bạn nhập không đúng định
dạng"

ControlToValidate="txtEmail"></asp:RegularExpressionValidator>
                </td>
            </tr>
            <tr>
                <td colspan="2">
```

```
                    <asp:Button ID="Button1" runat="server"
Text="Accept" />
                </td>
            </tr>
        </table>

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

Kết xuất của nó:




## VII. Tạo các điều khiển kiểm tra tính hợp lệ theo ý mình.

Trong phần này bạn sẽ học tạo các điều khiển tuỳ biến như thế nào. Chúng ta sẽ tạo  điều khiển. đầu tiên sẽ tạo một điều khiển kiểm tra độ dài của một trường nhập liệu trên Form LengthValidator, Bạn tạo mới một Control Kiểm tra tính hợp lệ bằng cách dẫn xuất từ lớp BaseValidator. Lớp BaseValidator là lớp cơ sở cho tất cả các điều khiển Validation, bao gồm RequiredFieldValidator và RegularExpressionValidator

Lớp cở sở là lớp 1 lớp phải được cài đặt mà yêu cầu bạn cài đặt một phương thức đơn.

- EvaluateIsValid: Trả về giá trị True khi trường kiểm tra tính hợp lệ trên Form là hợp lệ.
- GetControlValidationValue: Cho phép bạn điền giá trị của điều khiển sẽ được kiểm tra tính hợp lệ.

Khi bạn tạo một điều khiển kiểm tra tính hợp lệ tuỳ biến, bạn override phương thức EvaluateIsValid() và trong phương thức này bạn gọi GetControlValidationValue để lấy giá trị của trường cần kiểm tra tính hợp lệ.

Tạo một Điều khiển LengthValidator.

Trong phần này bạn sẽ được học cách tạo một điều khiển đơn giản để kiểm tra độ dài của trường nhập liệu.

Dưới đây mà mã nguồn của lớp LengthValidator.cs

Code 10 LenghtValidator.cs

```
using System;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace myControls
{
    public class LengthValidator : BaseValidator
    {
        int _maximumLength = 0;
        public int MaximumLength
        {
            get { return _maximumLength; }
            set { _maximumLength = value; }
        }
        protected override bool EvaluateIsValid()
        {
            String value =
this.GetControlValidationValue(this.ControlToValidate);
            if (value.Length > _maximumLength)
                return false;
            else
                return true;
        }
    }
}
```

Trong Lớp đoạn mã trên ta thấy LengthValidator được kế thừa từ lớp BaseValidator, trong lớp mới này ghi đề phương thức EvaluateIsValid. Giá trị của điều khiển được kiểm chứng được lấy về với phương thức GetControlValidationValue Và độ dài của giá trị sẽ được so sánh với thuộc tính MaximumLength.

Đế sử dụng điều khiển LenghtValidator chúng ta phải đăng ký điều khiển này ở đầu trang với thẻ chỉ dẫn <%@ Register %>. nếu bạn muốn sử dụng LengthValidator cho nhiều trang bạn có thể đăng ký điều khiển này ở phần <Page> trong file Web configuration.

Ví dụ sử dụng LengthValidator

Code 11

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="LengthValidator.aspx.cs" Inherits="LengthValidator" %>
<%@ Register TagPrefix="validator" Namespace="myControls" %>
```

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Use LenghtValidator</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="Label1" runat="server" Text="Ghi
chú"></asp:Label>
        <br />
        <asp:TextBox ID="txtghichu" runat="server" Height="67px"
TextMode="MultiLine"
            Width="176px"></asp:TextBox>
         <validator:LengthValidator ID="validatorLength"
ControlToValidate="txtghichu" runat="server" ErrorMessage="Phải nhỏ hơn
20 ký tự" MaximumLength="20"></validator:LengthValidator>
        <br />
        <asp:Button ID="Button1" runat="server" Text="Accept"
Width="85px" />
    </div>
    </form>
</body>
</html>
```

Chỉ dẫn <%@ Register %>, Với tiền tố TagPrefix=”Validator” và Chỉ đến Namespace=”MyControls” và sử dụng điều khiển này giống với các điều khiển Validator khác.

# Chương 4. Sử dụng các điều khiển khác.

Ở phần đầu của chương này các bạn sẽ được học sử dụng điều khiển FileUpload để cho phép chúng ta đưa các file dữ liệu lên Server, như là các file ảnh, word hay excel…

Ở chương này các bạn cũng được học các điều khiển khác như Calendar, AdRotator, Multiview, Wizard.

## I. File Upload.

Điều khiển FileUpload cho phép người sử dụng Upload file từ chính ứng dụng Web của mình.

File sau khi Upload có thể lưu trữ ở 1 nơi nào đó có thể là trên ổ cứng hay trong Database.

Điều khiển FileUpload hỗ trợ các thuộc tính sau:

| Thuộc tính | Ý nghĩa |
|---|---|
| Enable | Cho phép bạn vô hiệu hoá điều khiển FileUpload. |
| FileBytes | Cho phép lấy nội dung file đã được upload như một mảng Byte. |
| FileContent | Cho phép lấy nội dung của file đã được upload theo dòng dữ liệu |
| FileName | Lấy tên file được Upload |
| HasFile | Trả về giá trị đúng khi File được Upload |
| PostedFile | Enables you to get the uploaded file wrapped in the HttpPostedFile object. |

Điều khiển FileUpload hỗ trợ các phương thức

- Focus: Enables you to shift the form focus to the FileUpload control.
- SaveAs: Cho phép bạn lưu file được upload lên hệ thống.

Thuộc tính PostedFile của điều khiển FileUpload cho phép lấy thông tin từ File upload được bao bọc trong đối tượng HttpPostedFile. đối tượng này sẽ đưa thêm thông tin về Upload file.

Lớp HttpPostedFile gồm các thuộc tính sau:

- ContentLength: Lấy về kích thước của File Upload tính theo byte
- ContentType: lấy kiểu MIME của File Upload
- FileName: cho phép lấy tên của file được upload.
- InputStream: Enables you to retrieve the uploaded file as a stream.

Lớp HttpPostedFile chỉ hỗ trợ phương thức

- SaveAs: Cho phép bạn lưu file được upload lên hệ thống.

**Upload 1 file lên server**

Trong ví dụ sau bạn sẽ được học cách Upload 1 file ảnh lên đĩa cứng của Server.

Code 1a. Fileupload.aspx

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

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

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

        <asp:Label ID="Label1" runat="server" Text="Chọn
File"></asp:Label>
        <asp:FileUpload ID="FileUpload1" runat="server" Width="286px"
/>
        <br />
        <asp:Button ID="Button1" runat="server" Text="Add image"
Width="92px"
            onclick="Button1_Click" />
        <hr />
        <br />
        <asp:DataList ID="listImage" RepeatColumns="3" runat="server">
         <ItemTemplate>
            <asp:Image ID="Image1" ImageUrl='<%# Eval("Name",
"~/Upload/{0}") %>' style="width:200px" Runat="server" />
            <br />
            <%# Eval("Name") %>
         </ItemTemplate>
        </asp:DataList>

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

Code 1b.Fileupload.aspx.cs

```
using System;
using System.Data;
using System.IO;

public partial class _Default : System.Web.UI.Page
{
```

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

    }

    protected void Page_PreRender()
    {
        string upload_folder = MapPath("~/Upload/");
        DirectoryInfo dir = new DirectoryInfo(upload_folder);
        listImage.DataSource = dir.GetFiles();
        listImage.DataBind();
    }

    bool CheckFileType(string fileName)
    {
        string ext = Path.GetExtension(fileName);
        switch (ext.ToLower())
        {
            case ".gif":
                return true;
            case ".png":
                return true;
            case ".jpg":
                return true;
            case ".jpeg":
                return true;
            default:
                return false;
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        if (FileUpload1.HasFile)
        {
            if(CheckFileType(FileUpload1.FileName))
            {
                string filepath = "~/Upload/" + FileUpload1.FileName;
                FileUpload1.SaveAs(MapPath(filepath));
            }
        }
    }
}
```

Giải thích ví dụ trên: Trong sự kiện Button1_Click Kiểm tra có tồn tại File để Upload? Nếu đúng thì kiểm tra kiểm tra file upload có phải đúng định dạng của ảnh không bằng hàm CheckFileType nếu đúng thì sẽ thực hiện việc ghi file lên server với phương thức SaveAs của điều khiển FileUpload.

## II. Điều khiển Calendar

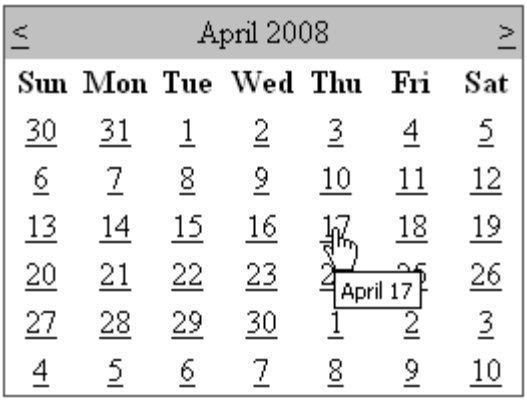
Bạn có thể hiển thị một lịch trên trang web của mình với điều khiển Calendar

Ví dụ sau sẽ trình bày một Calendar đơn giản

Code 2.

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

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Calendar</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
       <asp:Calendar ID="Calendar1" runat="server"></asp:Calendar>
    </div>
    </form>
</body>
</html>
```

Kết xuất của nó sẽ như sau:




Các thuộc tính:

| Thuộc tính | Ý nghĩa |
|---|---|
| DayNameFormat | Cho phép bạn chỉ rõ sự xuất hiện các ngày của tuần, có thể là các giá trị: FirstLetter, FirstTwoLetters, Full, Short, và Shortest. |
| NextMonthText | Chỉ định Text xuất hiện cho liên kết tháng tiếp theo |
| NextPrevFormat | Cho phép bạn chỉ rõ định dạng tháng tiếp theo và tháng trước đó, nó có thể là các giá trị: CustomText, FullMonth, và ShortMonth. |
| PreMonthText | Cho phép bạn chỉ định hiển thị text cho liên kế tháng trước đó |
| SelectedDate | Cho phép bạn lấy về hoặc thiết đặt cho ngày lựa chọn |
| SelectedDates | Cho phép bạn gán hoặc lấy về một tập các ngày được lựa chọn |

| | |
|---|---|
| SelectionMode | Cho phép chỉ định các ngày được lựa chọn có giá trị như thế nào, nó có thể là các giá trị sau: Day, DayWeek, DayWeekMonth, và none. |
| SelectMonthText | Cho phép hiển thị Text cho 1 tháng được chọn |
| SelectWeekText | Cho phép hiển thị Text cho 1 tuần được chọn |
| ShowdayHeader | Cho phép hiển thị tên ngày hay không trên đỉnh của điều khiển. |
| ShowNextPrevMonth | Cho phép hiển thị hay không liên kết đến tháng tiếp theo hoặc tháng trước đó. |
| ShowTitle | Cho phép bạn ẩn hay hiện Text trên thanh tiêu đề của điều khiển Calendar. |
| TitleFormat | Cho phép định dang text trên thanh tiêu đề, các giá trị của nó có thể là Month và MonthYear. |
| TodaysDate | Cho phép bạn chỉ rõ ngày hiện tại mặc định lấy ngày hiện tại trên Server. |

Các Sự kiện.

- DayRender: Raised as each day is rendered.
- SelectionChanged: Xảy ra khi một ngày mới, tuần mới hay tháng mới được lựa chọn.
- VisibleMonthChanged: xảy ra khi liên kết đến tháng tiếp theo hoặc tháng trước đó được nhấn.

Ví dụ sau sẽ trình bày cách lấy thông tin khi chúng ta lựa chọn nhiều ngày trên đối tượng Calendar.

Code 3a. Calendarmultiselect.aspx

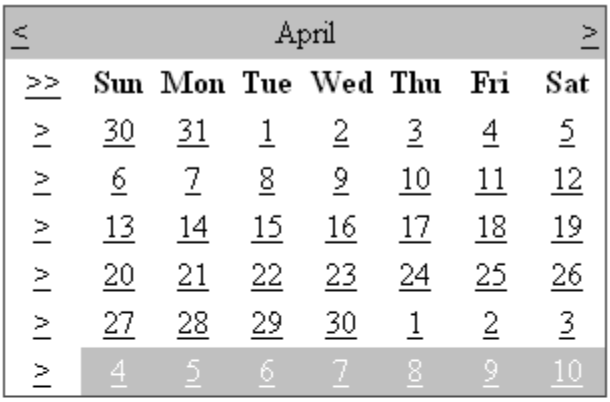
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Calendarmultiselect.aspx.cs"
Inherits="Calendarmultiselect" %>

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

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

```
    <form id="form1" runat="server">
    <div>
       <asp:Calendar ID="calendar1" SelectionMode="DayWeekMonth"
runat="server" SelectedDate="2008-04-18"
TitleFormat="Month"></asp:Calendar>
        <br />
         <asp:Button ID="btnSubmit" runat="server"
OnClick="btnSubmit_Click" Text="Submit"
            Width="83px" /><br />
        <hr />
    </div>
        <asp:BulletedList ID="bllresult" runat="server"
DataTextFormatString="{0:d}">
        </asp:BulletedList>
    </form>
</body>
</html>
```

Code 3b. Calendarmultiselect.aspx.cs

```
using System;

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

    }
    protected void btnSubmit_Click(object sender, EventArgs e)
    {
        bllresult.DataSource = calendar1.SelectedDates;
        bllresult.DataBind();
    }
}
```

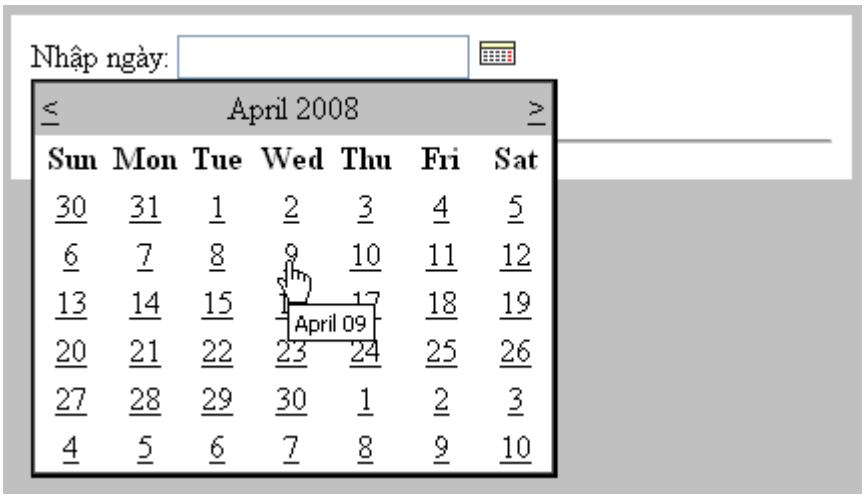
Kết xuất của ví dụ 3




- 5/4/2008
- 5/5/2008
- 5/6/2008
- 5/7/2008
- 5/8/2008
- 5/9/2008
- 5/10/2008

Tạo một Popup Datepicker

Trong phần này bạn sẽ học cách sử dụng điều khiển Calendar kết hợp với Javacript để tạo một Pop-up Date picker như trong hình dưới đây:




Code 4. popupdatepicker.aspx

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

<script runat="server">
```

```
    protected void calEventDate_SelectionChanged(object sender,
EventArgs e)
    {
        txtEventDate.Text = calEventDate.SelectedDate.ToString("d");
    }
    protected void btnSubmit_Click(object sender, EventArgs e)
    {
        lblResult.Text = "Bạn chọn: " + txtEventDate.Text;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Pop-up Date Picker</title>
    <script type="text/javascript">
    function displayCalendar()
    {
        var datePicker = document.getElementById('datePicker');
        datePicker.style.display = 'block';
    }
    </script>
    <style type="text/css">
        #datePicker
        {
            display:none;
            position:absolute;
            border:solid 2px black;
            background-color:white;
        }
        .content
        {
            width:400px;
            background-color:white;
            margin:auto;
            padding:10px;
        }
        html
        {
            background-color:silver;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">

      <div class="content">
        <asp:Label id="lblEventDate" Text="Nhập ngày:" Runat="server"
/>
        <asp:TextBox id="txtEventDate" Runat="server" />
        <img src="img.gif" alt="" onclick="displayCalendar()" />
        <div id="datePicker">
            <asp:Calendar id="calEventDate"
OnSelectionChanged="calEventDate_SelectionChanged" Runat="server" />
        </div>
        <br />
```

```
        <asp:Button id="btnSubmit" Text="Submit" Runat="server"
OnClick="btnSubmit_Click" />
        <hr />
        <asp:Label id="lblResult" runat="server" />
    </div>
    </form>
</body>
</html>
```

## III. Điều khiển Advertisements(Trình bày quảng cáo - Adrotator)

Cho phép bạn hiển thị các ảnh quảng cáo khác nhau trong 1 trang. bạn có thể lưu trữ các quảng cáo của bạn trong 1 file XML hoặc một bảng cơ sở dữ liệu.

. Các thuộc tính

| Thuộc tính | Ý nghĩa |
|---|---|
| AdvertisementFile | Cho phép bạn chỉ định đến đường dẫn file XML chứa đựng danh sách ảnh quảng cáo |
| AlternateTextField | Cho phép chỉ định đến tên của trường để hiển thị nội dung thay thế khi ảnh quảng cáo vì một lý do nào đó không hiển thị được |
| DataMenber | Cho phép ràng buộc đến một thành viên cơ sở dữ liệu của nguồn cơ sở dữ liệu |
| DataSource | Chỉ  rõ cở sở dữ liệu chứa danh sách các banner quảng cáo |
| DataSourceID | Ràng buộc đến 1 cơ sở dữ liệu |
| ImgUrlFile | Chỉ rõ đến trường chứa đường dẫn banner quảng cáo |
| KeywordFilter | Cho phép bạn lọc quảng cáo bởi 1 từ khoá |
| NavigateUrlField | Chỉ rõ đến tên của trường chứa các liên kết quảng cáo |
| Target | Cho phép bạn mở ra một cửa sổ mới khi nhấn vào banner quảng cáo |

Sự kiện

- AdCreated: Xảy ra sau khi điều khiển Adrotator lựa chọn một quảng cáo nhưng trước khi điều khiển AdRotator đưa ra quảng cáo.

Chú ý rằng điều khiển AdRotator chứa đựng thuộc tính KeywordFilter bạn có thể cung cấp mỗi banner quảng cáo với 1 từ khoá sau đó khi trình bày quảng cáo chúng ta có thể lọc những quảng cáo theo điều kiện để hiển thị.

Ví dụ như trên trang của bạn, bạn cần trình bầy quảng cáo ở 4 vị trí trên banner(top) của trang, bên trai, phải và phía chân(bottom) của trang. Khi đó bạn gán với mỗi ảnh quảng cáo một từ khoá(ví dụ tương ứng với 1- top, 2- right, 3 –

bottom, 4 - left)  và khi trình bầy trong điều khiển AdRotator ta có thể dùng thuộc tính KeywordFilter để lọc và trình bày.

Ví dụ 1: Trên trang web của bạn trình bầy quảng cáo ở hai vị trí(trên đầu và bên phải của trang) layout như sau


Code 5a: AdRotatorXML.aspx

```
<%@ Page Language="VB" AutoEventWireup="false"
CodeFile="AdRotatorXML.aspx.vb" Inherits="_Default" %>

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>AdRotator XML</title>
    <style type="text/css">
      #wrapper {width: 782px; margin: 0 auto;}
      #border { border: 1px solid #8ECE5F; margin: 20px 0px 5px 0px;
background-color: #ffffff; min-height: 776px; float: right; width:
780px;}
      #header {background-color:#BECEF5; text-align: width:782px;
height:100px; center; margin: 6px 0px 0px 0px;}
      #right{background-color:#BECEF5; width:170px;margin: 6px 0px 0px
4px; height:300px; float:right}
      .box{ float:right;padding:10px; border-left:solid 1px black;}
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div id="wrapper" >
     <div id="border">
```

```
      <div id="header">
          <asp:AdRotator ID="advbanner"
AdvertisementFile="~/App_Data/AdList.xml" KeywordFilter="banner"
runat="server" />
      </div>

      <div id="right">
        <div class="box">
        <asp:AdRotator ID="AdRotator1"
AdvertisementFile="~/App_Data/AdList.xml" KeywordFilter="right"
runat="server" />
        </div>
        <div class="box">
        <asp:AdRotator ID="AdRotator2"
AdvertisementFile="~/App_Data/AdList.xml" KeywordFilter="right"
runat="server" />
        </div>
      </div>
     </div>
    </div>
    </form>
</body>
</html>
```

Nội dung File XML

Code 5b.

```
<?xml version="1.0" encoding="utf-8" ?>
<Advertisements>
  <Ad>
    <ImageUrl>~/Advertisement/images/banner.png</ImageUrl>
    <Width>782</Width>
    <Height>100</Height>
    <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
    <AlternateText>Diễn đàn ĐH Hoa Sen</AlternateText>
    <Impressions>50</Impressions>
    <Keyword>banner</Keyword>
  </Ad>
  <Ad>
    <ImageUrl>~/Advertisement/images/banner2.gif</ImageUrl>
    <Width>782</Width>
    <Height>100</Height>
    <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
    <AlternateText>Diễn đàn ĐH Hoa Sen</AlternateText>
    <Impressions>50</Impressions>
    <Keyword>banner</Keyword>
  </Ad>
  <Ad>
    <ImageUrl>~/Advertisement/images/anh1.gif</ImageUrl>
    <Width>150</Width>
    <Height>150</Height>
    <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
    <AlternateText>Box Advertisement 1</AlternateText>
    <Impressions>50</Impressions>
```

```
  <Keyword>right</Keyword>
</Ad>
<Ad>
  <ImageUrl>~/Advertisement/images/anh1.gif</ImageUrl>
  <Width>150</Width>
  <Height>150</Height>
  <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
  <AlternateText>Box Advertisement 1</AlternateText>
  <Impressions>50</Impressions>
  <Keyword>right</Keyword>
</Ad>
<Ad>
  <ImageUrl>~/Advertisement/images/anh2.gif</ImageUrl>
  <Width>150</Width>
  <Height>150</Height>
  <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
  <AlternateText>Box Advertisement 1</AlternateText>
  <Impressions>50</Impressions>
  <Keyword>right</Keyword>
</Ad>
<Ad>
  <ImageUrl>~/Advertisement/images/anh3.gif</ImageUrl>
  <Width>150</Width>
  <Height>150</Height>
  <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
  <AlternateText>Box Advertisement 1</AlternateText>
  <Impressions>50</Impressions>
  <Keyword>right</Keyword>
</Ad>
<Ad>
  <ImageUrl>~/Advertisement/images/anh4.gif</ImageUrl>
  <Width>150</Width>
  <Height>150</Height>
  <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
  <AlternateText>Box Advertisement 1</AlternateText>
  <Impressions>50</Impressions>
  <Keyword>right</Keyword>
</Ad>
<Ad>
  <ImageUrl>~/Advertisement/images/anh5.gif</ImageUrl>
  <Width>150</Width>
  <Height>150</Height>
  <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
  <AlternateText>Box Advertisement 1</AlternateText>
  <Impressions>50</Impressions>
  <Keyword>right</Keyword>
</Ad>
<Ad>
  <ImageUrl>~/Advertisement/images/anh6.gif</ImageUrl>
  <Width>150</Width>
  <Height>150</Height>
  <NavigateUrl>http://www.sinhvienhoasen.com</NavigateUrl>
  <AlternateText>Box Advertisement 1</AlternateText>
  <Impressions>50</Impressions>
  <Keyword>right</Keyword>
</Ad>
 </Advertisements>
```

Cách thực hiện bạn đưa XML có cấu trúc như code 5b, sau đó trong trang AdRotatorXML.aspx bạn đưa điều khiển AdRotator vào và đặt cho nó hai thuộc tính AdvertisementFile chỉ đến File XML bạn vừa tạo, và thuộc tính KeywordFilter theo thẻ Keyword trong file XML.

## IV. Điều khiển hiển thị các trang khác nhau

Điều khiển MultiView cho phép bạn ẩn hoặc hiện các phần khác nhau của trang Web, điều khiển này tiện ích khi bạn tạo một TabPage. Nó thực sự tiện ích khi bạn muốn chia 1 trang web có độ dài lớn thành các phần để hiển thị

Điều khiển MultiView chứa đựng 1 hoặc nhiều điều khiển View, bạn sử dụng Multiview để lựa chọn các điều khiển View để trình bày.

Điều khiển MultiView hỗ trợ các thuộc tính.

- ActiveViewIndex: Lựa chọn điều khiển View được đưa ra hiển thị bằng chỉ số Index
- Views: Cho phép bạn lấy về tập hợp các điều khiển View chứa đựng trong điều khiển MultiView.

Điều khiển MultiView hỗ trợ hai phương thức.

- GetActiveView: Cho phép lấy về thông tin của điều khiển View được lựa chọn.
- SetActiveView: cho phép bạn thiết lập điều khiển View được hiên thị.
- Và MultiView hỗ trợ sự kiện sau:
- ActiveViewChanged: Xảy ra khi điều khiển View được lựa chọn

**Cách sử dụng**

### *1. hiển thị như một TabPage*

Khi bạn sử dụng MultiView kết hợp với điều khiển Menu bạn có thể tạo một TabPage

Ví dụ sau sẽ hướng dẫn bạn tạo một TabPage từ 2 điều khiển Menu và MultiView

Code 6a.

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

<script runat="server">

    void Menu1_MenuItemClick(object sender, MenuEventArgs e)
    {
        int index = Int32.Parse(e.Item.Value);
        MultiView1.ActiveViewIndex = index;
    }
```

```
    void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            MultiView1.ActiveViewIndex = 0;
        }
    }

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Create a TabView</title>
    <style type="text/css">
        html
        {
            background-color:silver;
        }
        .tabs
        {
            position:relative;
            top:1px;
            left:10px;
        }
        .tab
        {
            border:solid 1px black;
            background-color:#eeeeee;
            padding:2px 10px;
        }
        .selectedTab
        {
            background-color:white;
            border-bottom:solid 1px white;
        }
        .tabContents
        {
            border:solid 1px black;
            padding:10px;
            background-color:white;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div>
            <asp:Menu ID="Menu1"
            OnMenuItemClick="Menu1_MenuItemClick" runat="server"
            Orientation="Horizontal" StaticMenuItemStyle-CssClass="tab"
            StaticSelectedStyle-CssClass="selectedTab" CssClass="tabs">
            <Items>
                <asp:MenuItem Text="Tab 1" Value="0"
Selected="true"></asp:MenuItem>
                <asp:MenuItem Text="Tab 2" Value="1"></asp:MenuItem>
                <asp:MenuItem Text="Tab 3" Value="2"></asp:MenuItem>
```

```
            </Items>
        </asp:Menu>
        <div class="tabContents">
            <asp:MultiView ID="MultiView1" runat="server">
                <asp:View ID="view1" runat="server">
                    Day la View 1<br />
                    Day la View 1<br />
                    Day la View 1<br />
                    Day la View 1<br />
                    Day la View 1<br />
                    Day la View 1<br />
                </asp:View>
                <asp:View ID="view2" runat="server">
                    Day la View 2<br />
                    Day la View 2<br />
                    Day la View 2<br />
                    Day la View 2<br />
                    Day la View 2<br />
                    Day la View 2<br />
                </asp:View>
                <asp:View ID="view3" runat="server">
                    Day la View 3<br />
                    Day la View 3<br />
                    Day la View 3<br />
                    Day la View 3<br />
                    Day la View 3<br />
                    Day la View 3<br />
                </asp:View>
            </asp:MultiView>
        </div>
    </div>
    </form>
</body>
</html>
```

Trong ví dụ trên Menu được kết hợp với CSS để tạo ra các trạng thái khi MenuItem được chọn(StaticSelectedStyle-CssClass="selectedTab") và ko được chọn(StaticMenuItemStyle-CssClass="tab").

### 2. *Hiển thị nhiều phần trên trang.*

Bạn có thể chia một Form có độ dài lớn thành các thành phần nhỏ hơn và hiển thị từng phần, bạn có thể sử dụng các điều khiển Button nằm trong điều khiển MultiView và khi Button được nhấn thì Multiview sẽ xử lý thay đổi hiển thị View khác.

Điều khiển MultiView hỗ trợ các điều khiển lệnh sau:

- NextView: MultiView sẽ kích hoạt điều khiển View tiếp theo
- PrevView: MultiView sẽ kích hoạt điều khiển View trước đó
- SwitchViewByID: MultiView sẽ kích hoạt View chỉ định bởi đối số của điều khiển Button
- SwitchViewByIndex: MultiView sẽ kích hoạt View chỉ định bởi đối số của điều khiển Button

Bạn có thể sử dụng các điều khiển lệnh như Button, ImageButton, LinkButton. Và thiết lập thuộc tính CommandName, với trường hợp điều khiển lệnh là SwitchViewByID và SwitchViewByIndex bạn thiết lập thêm thuộc tính CommandArgument.

Ví dụ sau sẽ hướng dẫn bạn tạo một Form có nhiều phần với việc sử dụng điều khiển lệnh NextView.

Code 7.

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

<script runat="server">
    void View3_Active(object sender, EventArgs e)
    {
        lblRHoten.Text = txtHoten.Text;
        lblRCMT.Text = txtCMT.Text;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>MultiPart View</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:MultiView ID="MultiView1" runat="server"
ActiveViewIndex="0">
            <asp:View ID="View1" runat="server">
                <asp:Label ID="lblHoten" runat="server" Text="Nhập họ
tên"></asp:Label>
                <asp:TextBox ID="txtHoten" runat="server"
Width="198px"></asp:TextBox><br />
                <asp:Button ID="btnNext1" runat="server"
CommandName="NextView" Text="Next" Width="98px" /></asp:View>
            <asp:View ID="View2" runat="server">
                <asp:Label ID="lblCMT" runat="server" Text="Nhập số
CMT"></asp:Label>
                <asp:TextBox ID="txtCMT" runat="server"></asp:TextBox>
                <br />
                <asp:Button ID="btnNext2" runat="server"
CommandName="NextView" Text="Next" Width="99px" /></asp:View>
            <asp:View ID="View3" OnActivate="View3_Active"
runat="server">
                <asp:Label ID="Label1" runat="server" Text="Họ
tên:"></asp:Label>
                <asp:Label ID="lblRHoten" runat="server"
Text="Label"></asp:Label><br />
                <asp:Label ID="Label2" runat="server" Text="Số
CMT:"></asp:Label>
                <asp:Label ID="lblRCMT" runat="server"
Text="Label"></asp:Label></asp:View>
```

```
        </asp:MultiView></div>
    </form>
</body>
</html>
```

Trong ví dụ trên 2 button đầu thiết lập thuộc tính CommandName có giá trị là NextView, do điều khiển MultiView hỗ trợ lệnh NextView lên khi nhấn vào 2 Button đó thì lệnh NextView được thực hiện và kích hoạt View tiếp theo đến View 3 với sự kiện OnActive gọi hàm View3_Active và thực hiện lấy về giá trị của hai TextBox và gán vào 2 Label tương ứng.

## V. Hiển thị với điều khiển Wizard

Điều khiển Wizard giống với điều khiển MultiView có thể dùng để chia một Form lớn thành nhiều phần nhỏ. Tuy nhiên nó sẽ có thêm một số thuộc tính mà MultiView không hỗ trợ.

điều khiển Wizard có thể chứa nhiều điều khiển WizardStep con, nhưng chỉ 1 WirardStep được hiển thị tại 1 thời điểm.

Các thuộc tính.

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

- HeaderTemplate: hiển thị thanh tiêu đề ở đầu của điều khiển Wizard
- SlideBarTemplate: Cho phép hiển thị SlideBar trong điều khiển Wizard
- StartNavigationTemplate: Cho phép hiển thị Navigation ở bước bắt đầu
- StepNavigationTemplate: cho phép hiển thị Navigation ở các bước à không phải bước bắt đầu và kết thúc.

Điều khiển Wizard hỗ trợ các phương thức:

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

Điều khiển Wizard hỗ trợ các sự kiện:

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

Một điều khiển Wizard chứa đựng một hoặc nhiều WizardStep để diễn tả các bước strong quá trình Wizard.

Các WizardStep hỗ trợ các thuộc tính:

- AllowReturn: Ngăn cản hay cho phép người sử dụng trả về bước này từ một bước khác.
- Name: tên của điều khiển WirardStep
- StepType:Cho phép bạn gán hay lấy về kiểu của WirardStep nó có thể là các giá trị sau: Auto, Finish, Start, Complete và Step.
- Title: lấy về hoặc gán tiêu đề của điều khiển WizardStep tiêu đề này được hiển thị ở Wizard Slidebar

- Wizard: cho phép bạn lấy thông tin điều khiển Wizard chứa trong WizardStep.
- Các Sự kiện trong WizardStep
- Activate: Xảy ra khi một WizardStep được kích hoạt
- DeActivate: xảy ra khi WizardStep khác được kích hoạt.

StepType là thuộc tính quan trọng nhất của Wizard, thuộc tính nào xác định WizardStep được đưa ra như thế nào, mặc định là Auto

Ví dụ:

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

<script runat="server">
    void Wizard1_FinishButtonClick(object sender,
WizardNavigationEventArgs e)
    {
        lblbiet.Text = txt1.Text;
        lblkhoahoc.Text = txt2.Text;
    }
</script>

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Wizard </title>
    <style type="text/css">
        .wizard{border:solid 1px black;font:14px
Verdana,width:800px;height:300px;}
        .header{color:gray;font:bold 18px Verdana,}
        .sideBar{background-color:#eeeeee;padding-
left:10px;width:200px; height:23px;}
        .sideBar a{text-decoration:none;}
        .step{padding:10px;}
</style>
</head>

<body>
    <form id="form1" runat="server">
    <div>

        <asp:Wizard ID="Wizard1" HeaderText="Wizard control"
OnFinishButtonClick="Wizard1_FinishButtonClick"
            CssClass="wizard" Width="800px" HeaderStyle-
CssClass="header"
            SideBarButtonStyle-CssClass="sideBar" SideBarStyle-
Width="200px" SideBarButtonStyle-Width="200px" StepStyle-
CssClass="step" runat="server">
            <StepStyle CssClass="step"></StepStyle>
```

```
            <WizardSteps>
                <asp:WizardStep ID="WizardStep1" runat="server"
title="Lời giới thiệu">

                    <asp:Label ID="Label1" runat="server"
                        Text="Chào mừng bạn đến với Diễn đàn ĐH Hoa
Sen"></asp:Label>

                </asp:WizardStep>
                <asp:WizardStep ID="WizardStep2" runat="server"
title="step 1">

                    <asp:Label ID="Label2" runat="server"
                        Text="Bạn biết đến Diễn đàn Sinh Viên Hoa Sen
như thế nào?"></asp:Label>
                    <br />
                    <asp:TextBox ID="txt1"
runat="server"></asp:TextBox>

                </asp:WizardStep>

                <asp:WizardStep ID="WizardStep3" runat="server"
StepType="Finish"  title="step 2">

                    <asp:Label ID="Label3" runat="server"
                        Text="Bạn thích khoa nào nhất ở ĐH Hoa
Sen"></asp:Label>
                    <asp:TextBox ID="txt2"
runat="server"></asp:TextBox>

                </asp:WizardStep>

                <asp:WizardStep ID="WizardStep4" runat="server"
StepType="Complete" title="Tổng kết">
                    <asp:Label ID="Label4" runat="server" Text="Bạn
biết đến Diễn Đàn Sinh Viên Hoa Sen qua:"></asp:Label>
                    <asp:Label ID="lblbiet" runat="server"
Text=""></asp:Label><br />
                    <asp:Label ID="Label5" runat="server" Text="Bạn
thích nhất khoa:"></asp:Label>
                    <asp:Label ID="lblkhoahoc" runat="server"
Text=""></asp:Label>
                </asp:WizardStep>

            </WizardSteps>
        </asp:Wizard>

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

Giải thích ví dụ:

Trên ví dụ ta đưa vào 4 WizardStep, chương trình sẽ thực hiện trình tự theo các bước và kết thúc với việc nhấn nút Finish ở WizardStep3(được thiết lập thuộc tính

StepType=’Finish’) thông qua hàm “Wizard1_FinishButtonClick” sau khi thực hiện hàm này nó sẽ chuyến sang và hiển thị nội dung công việc cần thực hiên thông 2 điều khiển label trên WizardStep4 và điều khiển này được thiết lập thuộc tính StepType=”Complete”.

# Chương 5 Thiết kế Website với MasterPage

Bạn đang gặp vấn đề thiết kế layout chung cho website của mình? MasterPage có thể là một đáp án tốt cho bạn giải quyết bài toán trên.

Với MasterPage để tạo một giao diện chung nhất cho Website của mình. Ví dụ website của bạn có layout gồm 5 phần banner, footer, left, right

Và content(phần chứa nội dung hiển thị cho các tin). Như vậy bạn có thể thấy trên trang web có các phần có cách trình bầy không thay đổi trong quá trình duyệt tin. banner, footer, left, right: bạn có thể để các phần này vào một MasterPage và tất các trang web trong website của bạn sẽ áp dụng layout này thông qua contentpage

## I. Tạo MasterPage

Các bước tạo MasterPage giống với tạo các trang aspx bình thường nhưng trên hộp Add New Item bạn chọn MasterPage bạn thấy dưới ô đặt tên phần mở rộng của nó sẽ là Master. Bạn có thể tạo nhiều MasterPage cho ứng dụng web của mình.

Ví dụ:

Code 1.

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeFile="MasterPage.master.cs" Inherits="MasterPage" %>

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>MasterPage</title>
    <style type="text/css">
        html{ background-color:#e5e5e5; font-family:Verdana;}
        .content{ background-color:White; border:black 1px solid;
width:700px; margin:auto;}
        .leftcolumm{ float:left; border-right:black 1px solid;
width:200px; padding:5px; height:300px}
        .rightcolumm{ float:left; padding:5px; height:300px}
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div class="content">
        <div class="leftcolumm">
            <asp:ContentPlaceHolder ID="contentplace1"
runat="server"></asp:ContentPlaceHolder>
        </div>

        <div class="rightcolumm">
            <asp:ContentPlaceHolder id="ContentPlaceHolder1"
runat="server">

            </asp:ContentPlaceHolder>
        </div>
    </div>
    </form>
```

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

Để sử dụng MasterPage cho các trang aspx bạn cần khai báo trên chỉ dẫn <%@ Page %>  với thuộc tính MasterPageFile=”Tên _MasterPage của bạn”

ví dụ

code 2

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master" %>
<asp:Content ID="contentleft" ContentPlaceHolderID="contentplace1"
runat="server">
    Trinh bay menu Left<br />
    Trinh bay menu Left<br />
    Trinh bay menu Left<br />
    Trinh bay menu Left<br />
</asp:Content>

<asp:Content ID="content1" ContentPlaceHolderID="ContentPlaceHolder1"
runat="server">
    Trinh bay noidung<br />
    Trinh bay noidung<br />
    Trinh bay noidung<br />
    Trinh bay noidung<br />
</asp:Content>
```

Trên trang aspx bạn khai báo 2 điều khiển asp:content và trong thuộc tính của nó bạn thiết lập ContentPlaceHolderID=”Ten_Placeholder trong trang masterPage”.

**Tạo một Content mặc định**

Bạn không thể kết hợp một điều khiển Content với tất cả các điều khiển ContentPlaceHolder. Khi trên trang MasterPage của bạn có 3 ContentPlaceHolder mà trong trang aspx của bạn chỉ sử dụng hai điều khiển aspContent và chỉ đến 2 ContentPlaceHolder trên trang MasterPage khi đó mặc định ContentPlaceHolder thứ 3 vẫn được hiển thị.

**Đăng ký Master Page trong WebConfiguration**

Bạn có thể áp dụng MasterPage cho tất cả các trang trong một Folder hay tất cả các Content trong ứng dụng web của mình.

Thay vì việc trong mỗi trang aspx bạn phải gọi lời chỉ dẫn trang thẻ chỉ dẫn <%@ Page %> thì bạn có thể thêm nó vào trong file webconfiguration

Trong ví dụ dưới đây là cách bạn sẽ đăng ký một trang SimpleMasterPage vào ứng dụng web của mình để có thể áp dụng cho tất cả các trang trong foder(Sub Folder) chứa

Chú ý rằng file web.config này phải cùng nằm trong foleder đó:

```
  <system.web>
    <pages masterPageFile="~/SimpleMasterPage.master" />
  </system.web>
```

**Sử dụng FindControl trong MasterPage.**

Khi bạn cần chỉnh sửa hay lấy thông tin của 1 điều khiển từ MasterPage bạn có thể sử dụng phương thức FindControl() trong một Content Page

Ví dụ

Trang Findcontrol.master

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeFile="Findcontrol.master.cs" Inherits="Findcontrol" %>

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Findcontrol</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="lblTitle" runat="server" /><br />
        <asp:ContentPlaceHolder id="ContentPlaceHolder1"
runat="server">

        </asp:ContentPlaceHolder>
    </div>
    </form>
</body>
</html>
```

Trang Findcontrol.aspx

```
<%@ Page Language="C#" MasterPageFile="~/Findcontrol.master"%>
<script runat="server">
    void Page_Load(object sender, EventArgs e)
    {
        Label lbl = (Label)Master.FindControl("lblTitle");
        lbl.Text = "Diễn đàn đại học Hoa Sen";
    }
</script>

<asp:Content ID="content" runat="server"
ContentPlaceHolderID="ContentPlaceHolder1">
 Lay gia tri tu masterpage<br />
 Lay gia tri tu masterpage<br />
 Lay gia tri tu masterpage<br />
 Lay gia tri tu masterpage<br />
 Lay gia tri tu masterpage<br />
</asp:Content>
```

Trình bày thuộc tính MasterPage

Bạn có thể trình bày thuộc tính , phương thức từ MasterPage và có thể chỉnh sửa thuộc tính hay phương thức từ trang Content.

Ví dụ

Trang Expose.master

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeFile="Expose.master.cs" Inherits="Expose" %>

<script runat="server">

    public string BodyTitle
    {
        get { return lblBody.Text; }
        set { lblBody.Text = value; }
    }

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Expose MasterPage Property</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <h1><asp:Label ID="lblBody" runat="server" Text="Sinh Viên Hoa
Sen"></asp:Label></h1>
        <asp:ContentPlaceHolder id="ContentPlaceHolder1"
runat="server">

        </asp:ContentPlaceHolder>
    </div>
    </form>
</body>
</html>
```

Trang Expose.aspx

```
<%@ Page Language="C#" MasterPageFile="~/Expose.master"
AutoEventWireup="true" %>
<%@ MasterType VirtualPath="~/Expose.master" %>
<script runat="server">
    void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            Master.BodyTitle = "Diễn Đàn Sinh Viên Hoa Sen";
        }
    }
</script>
<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">
 Property, Property, Property, Property, Property, <br />
 Property, Property, Property, Property, Property, <br />
 Property, Property, Property, Property, Property,<br />
 Property, Property, Property, Property, Property<br />
</asp:Content>
```

Trong ví dụ trên bạn sẽ thấy một điểm mới đó là chỉ dẫn MasterType, để sử dụng được thuộc tính hay phương thức từ MasterPage ta phải thêm vào chỉ dẫn MasterType và đặt thuộc tính VirtualPath=”Ten_Mastepage”.

# Chương 6. Thiết kế Website với themes

Bạn có thể tạo Themes bằng cách  bạn thêm vào ứng dụng một Folder đặc biệt của ứng dụng được đặt tên là App_Themes trong thư mục gốc của ứng dụng,

Trong Folder themes có thể chứa nhiều kiểu File bao gồm cả ảnh và text. bạn có thể tổ chức nội dung của themes trong các folder con của nó. Hai kiểu của File quan trọng nhất trong themes là

- skin files
- Cascading Style Sheet files

Trong chương này bạn sẽ được học cả hai cách trình bày trong thêm với CSS và Skin

## 1. Thêm Skin vào trong themes

Một Themes có thể chứa đựng một hoặc nhiều Skin, một Skin cho phép bạn thay đổi một số thuộc tính của các điều khiển trong asp.net mà những thuộc tính đó biểu diễn hiển thị của control đó với người dùng.

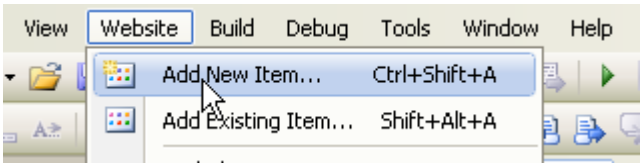
Bạn có thể tạo ra các skin mặc định(sẽ áp dụng cho các điều khiển trên form), skin áp dụng cho 1 điều khiển được chỉ định form.

Để không áp dụng skin mặc định cho điều khiển bạn có thể sử dụng thuộc tính EnableTheming=”false”.

Ví dụ:

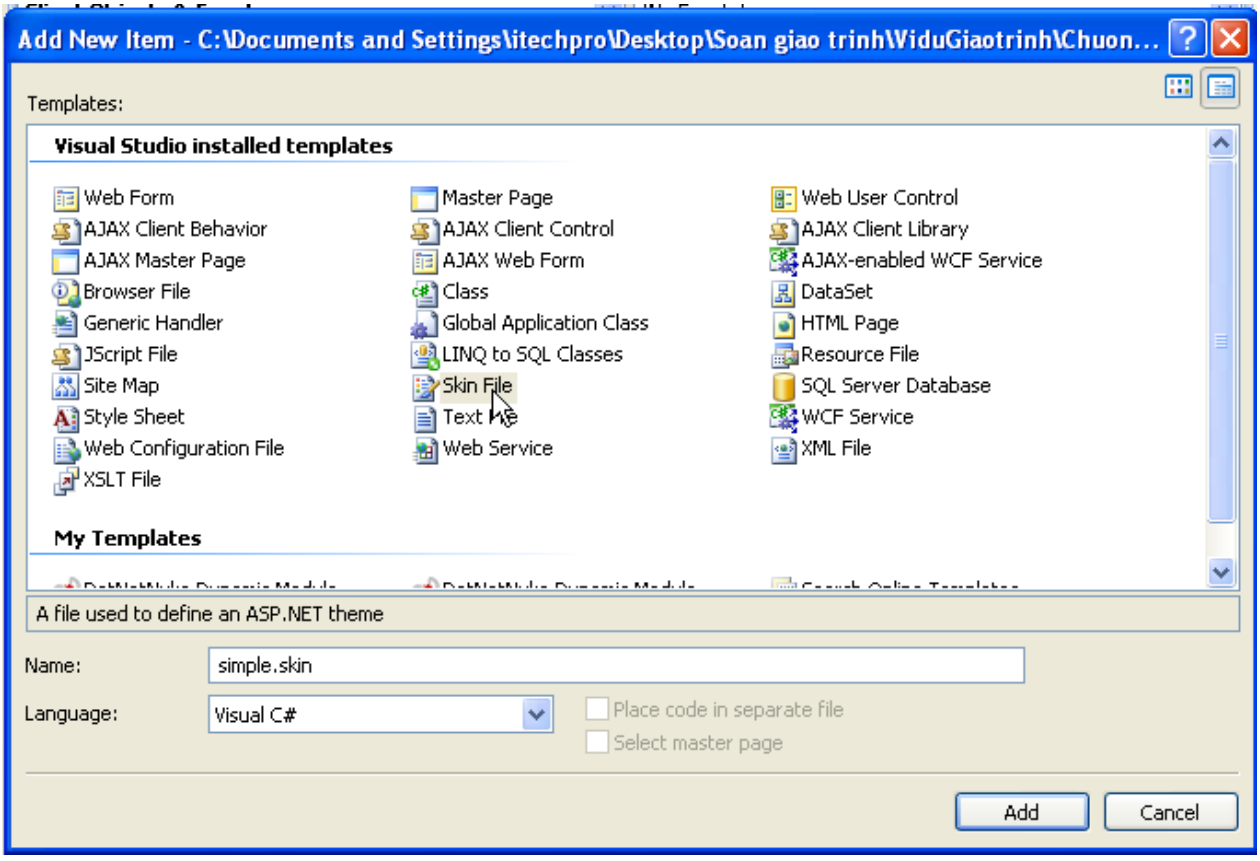
Trong ví dụ này tôi sẽ đưa ra 3 trường hợp(mặc định, chỉ định, ko áp dụng) áp dụng Skin vào trong các điều khiển trên Form.

- bước 1: Bạn chọn như hình 1



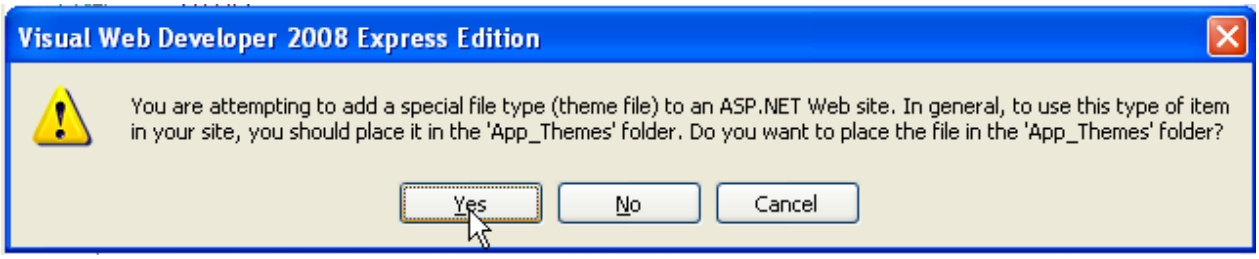

Hình 1

- bước 2: hộp thoại Add New Item hiện ra bạn chọn Skin File và đặt tên Skin của mình vào hộp Name và nhấn vào nút Add như hình sau:

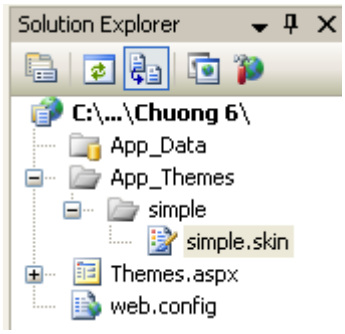



Hình 2

- bước 3: hộp thoại thông báo hiện ra bạn chọn YES.




- bước 4: trong Project của bạn sẽ thêm vào thư mục App_Theme và Skin "Simple" sẽ có đường dẫn sau:.




Trong File simple.skin bạn soạn nội dung như sau:

```
<asp:TextBox SkinID="txtChidinh" BorderStyle="dashed" BorderWidth="5px"
Runat="Server" />
<asp:TextBox BorderStyle="none" BorderWidth="0px" Runat="Server" />
```

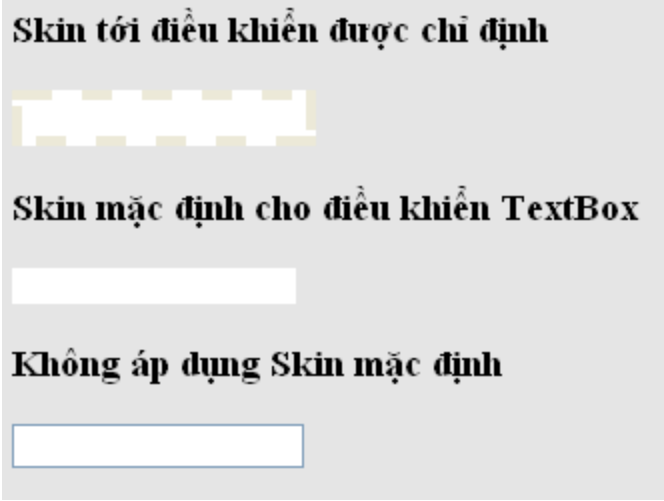
Và để sử dụng skin này cho trang aspx của chúng ta, bạn phải thêm vào chỉ dẫn Theme=”simple” trong chỉ dẫn <%@ Page …%>

Như trang Themes.aspx sau:

Code1.

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

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Using Themes</title>
</head>
<body style="background-color:#e5e5e5;">
    <form id="form1" runat="server">
    <div>
    <h3>Skin tới điều khiển được chỉ định</h3>
     <asp:TextBox ID="txtChidinh" runat="server"
SkinID="txtChidinh"></asp:TextBox>
    <h3>Skin mặc định cho điều khiển TextBox</h3>
     <asp:TextBox ID="txtmacdinh" runat="server"></asp:TextBox>
    <h3>Không áp dụng Skin mặc định</h3>
     <asp:TextBox ID="txtnone" runat="server"
EnableTheming="false"></asp:TextBox>
    </div>
    </form>
</body>
</html>
```

kết xuất của chương trình:




Khi sử dụng Themes thì mặc định nếu trong các điều khiển trên Form có quy định thuộc tính thì hiển thị của điều khiển đó sẽ bị ảnh hưởng của các thuộc tính đó. Tuy nhiên bạn cũng có thể ghi đè các thuộc tính đó để sử dụng thuộc tính trong Themes quy định, bằng cách đưa vào chỉ dẫn <%@ Page StyleSheetTheme=”simple” %>.

**Đăng ký Themes với web.config**

Khi bạn muốn sử dụng themes cho tất cả các trang web trong website của mình bạn có thể đăng ký nó vào trong file web.config trong ứng dụng web của mình như sau:

```
<configuration>
     <system.web>
          <pages theme=”simple” />
      </system.web>
</configuration>
```

Đ ghi đè vào thuộc tính của điều khiển trong website

```
<configuration>
     <system.web>
          <pages styleSheetTheme=”simple” />
      </system.web>
</configuration>
```

Trong một trang ta ko muốn sử dụng Themes, ta có thể gỡ bỏ nó trong trang này bằng cách: <%@ Page Language=”C#” EnableTheming=”false” %>

Thêm CSS tới Themes

Css là một thay thế cho skins để điều khiển cách xuất hiện của các thành phần của cả HTML và ASPNET

Khi thêm css vào folder Themes thì nó sẽ được áp dụng cho tất cả các trang được áp dụng theme

Ví dụ:

File simple.css

```
html{background-color:gray;font:14px Georgia,Serif;}
.content{margin:auto;width:600px;border:solid 1px black;background-
color:White;padding:10px;}
h1{color:Gray;font-size:18px;border-bottom:solid 1px orange;}
label{font-weight:bold;}
input{background-color:Yellow;border:double 3px orange;}
.button{background-color:#eeeeee;}
```

File skintotheme.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>SIMPLE CSS</title>
</head>
<body>
    <form id="form1" runat="server">
        <div class="content">
            <h1>Registration Form</h1>
            <asp:Label id="lblFirstName" Text="First Name:"
AssociatedControlID="txtFirstName" Runat="server" /><br />
            <asp:TextBox id="txtFirstName" Runat="server" /><br /><br
/>
            <asp:Label id="lblLastName" Text="Last Name:"
AssociatedControlID="txtLastName" Runat="server" /><br />
            <asp:TextBox id="txtLastName" Runat="server" /><br /><br />
            <asp:Button id="btnSubmit" Text="Submit Form"
CssClass="button" Runat="server" />
        </div>
    </form>
</body>
</html>
```

Kết xuất của chương trình


Ở ví dụ trên css được sử dụng tới kiểu của các thành phần html, bởi vì điều khiển asp.net đưa ra mã html khi trình duyệt lên áp dụng css cho các điều khiển của aspx phải sử dụng với thẻ của HTML, như trog ví dụ trên cả hai điều khiển TextBox và Button được đưa ra mã HTML là thẻ <imput>.

**Đưa nhiều CSS vào 1 Themes**

Khi bạn muốn đưa nhiều css vào 1 thư mục themes, ví dụ bạn tạo ra hai file css là ThemeA.css và ThemesB.css thì khi sử dụng bạn phải dùng cú pháp:

<link href=”App_Themes/Simple/ThemeA.css” type=”text/css” rel=”stylesheet” />
<link href=”App_Themes/Simple/ThemeB.css” type=”text/css” rel=”stylesheet” />

# Chương 7 xây dựng và sử dụng các điều khiển do người dùng tạo ra

Một webcontrol cho phép bạn xây dựng những control mới từ những control của asp.net, bạn có thể dễ dàng mở rộng ASPNET Framework từ những control do mình tạo ra.

Việc thao tác với webcontrol thì hầu hết là giống với aspx, để sử dụng webcontrol trên trang aspx bạn phải đăng ký nó trước khi sử dụng với chỉ dẫn <%@ Register %> ví dụ

```
<%@ Register TagPrefix="main" TagName="cal" Src="~/Caculator.ascx" %>
```

TagPrefix: Chỉ định không gian tên mà bạn muốn kết hợp usercontrol và trang hiện tại

TagName: Chỉ định tên mà bạn muốn kết hợp usercontrol và trang hiện tại

Src: chỉ đến đường dẫn của control(.ascs)

Ví dụ 1: tạo một webcontrol thực hiện phép tính toán và sử dụng nó trong 1 trang aspx

Code 1: Caculator.ascx

```
<%@ Control Language="C#" AutoEventWireup="true"
CodeFile="Caculator.ascx.cs" Inherits="Caculator" %>
<asp:Label ID="Label1" runat="server" Text="Nhập a:"></asp:Label>
<asp:TextBox ID="txta" runat="server"></asp:TextBox>
 <asp:Label ID="Label2" runat="server" Text="Nhập b:"></asp:Label>
<asp:TextBox ID="txtb" runat="server"></asp:TextBox>
<hr />
<asp:Button ID="btnsum" OnCommand="Calculator" CommandName="cal"
CommandArgument="sum" runat="server" Text="+" Width="45px" />
<asp:Button ID="btnsub" OnCommand="Calculator" CommandName="cal"
CommandArgument="sub" runat="server" Text="-" Width="45px" />
<asp:Button ID="btnmul" OnCommand="Calculator" CommandName="cal"
CommandArgument="mul" runat="server" Text="X" Width="45px" />
<asp:Button ID="btndiv" OnCommand="Calculator" CommandName="cal"
CommandArgument="div" runat="server" Text="/" Width="45px" />
<hr />
<asp:Label ID="lblresult" runat="server" Text="Label"></asp:Label>
<hr />
<asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
    ControlToValidate="txta" ErrorMessage="Bạn phải nhập a"
Visible="False"></asp:RequiredFieldValidator>
<asp:RangeValidator ID="RangeValidator1" runat="server"
    ControlToValidate="txta" ErrorMessage="a phải là kiểu nguyên"
    MaximumValue="9999999" MinimumValue="0" Type="Integer"
Visible="False"></asp:RangeValidator>
<asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server"
    ControlToValidate="txtb" ErrorMessage="Bạn phải nhập b"
Visible="False"></asp:RequiredFieldValidator>
```

```
<asp:RangeValidator ID="RangeValidator2" runat="server"
    ControlToValidate="txtb" ErrorMessage="a phải là kiểu nguyên"
    MaximumValue="9999999" MinimumValue="0" Type="Integer"
Visible="False"></asp:RangeValidator>
<p>
     </p>
<asp:ValidationSummary ID="ValidationSummary1" runat="server" />
```

Code 2 Caculator.ascx.cs

```
using System;

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

    }

    protected void Calculator(object sender, CommandEventArgs e)
    {
        if (e.CommandName == "cal")
        {
            switch (e.CommandArgument.ToString())
            {
                case "sum":
                    lblresult.Text =
Convert.ToString(int.Parse(txta.Text) + int.Parse(txtb.Text));
                    break;
                case "sub":
                    lblresult.Text =
Convert.ToString(int.Parse(txta.Text) - int.Parse(txtb.Text));
                    break;
                case "mul":
                    lblresult.Text =
Convert.ToString(int.Parse(txta.Text) * int.Parse(txtb.Text));
                    break;
                case "div":
                    {
                        if (int.Parse(txtb.Text) != 0)
                            lblresult.Text =
Convert.ToString(int.Parse(txta.Text) / int.Parse(txtb.Text));
                    }
                    break;
                default:
                    //
                    break;
            }
        }
    }
}
```

Code 3: Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default.aspx.cs" Inherits="_Default" %>
<%@ Register TagPrefix="main" TagName="cal" Src="~/Caculator.ascx" %>
```

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Su dung Webcontrol</title>
    <style type="text/css">
        html{ background-color:#e5e5e5}
        .divmain{ background-color:White; margin:15px; padding:15px;
top:30px; left:150px; position:absolute;}
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div>
     <div class="divmain">
        <main:cal ID="maincal" runat="server" />
     </div>
    </div>
    </form>
</body>
</html>
```

Sử dụng chỉ dẫn Reference

Khi bạn load 1 Usercontrol với phương thức Page.LoadControl(), UserControl được trả về như một thực thể của lớp System.Web.UI.WebControl. có nghĩa là bao gồm tất cả các thuộc tính tuỳ biến, thuộc tính này không có sẵn như với trường hợp Load UserControl động.

Nếu bạn sử dụng Load UserControl động bạn cần ép kiểu của UserControl về đúng kiểu trước khi sử dụng các thuộc tín. Để lấy một tham chiếu tới một Usercontrol bạn cần  phải sử dụng chỉ dẫn <%@ Reference %>

Ví dụ: bạn muốn thăm dò ý kiến của mọi người xem họ đã từng sử dụng ngôn ngữ asp cơ bản hay ngôn ngữ asp.net ? Và tuỳ thuộc vào mỗi người họ sẽ chọn asp hay aspnet để đưa ra hiển thị một UserControl.

# Chương 8 Điều khiển ADO.NET

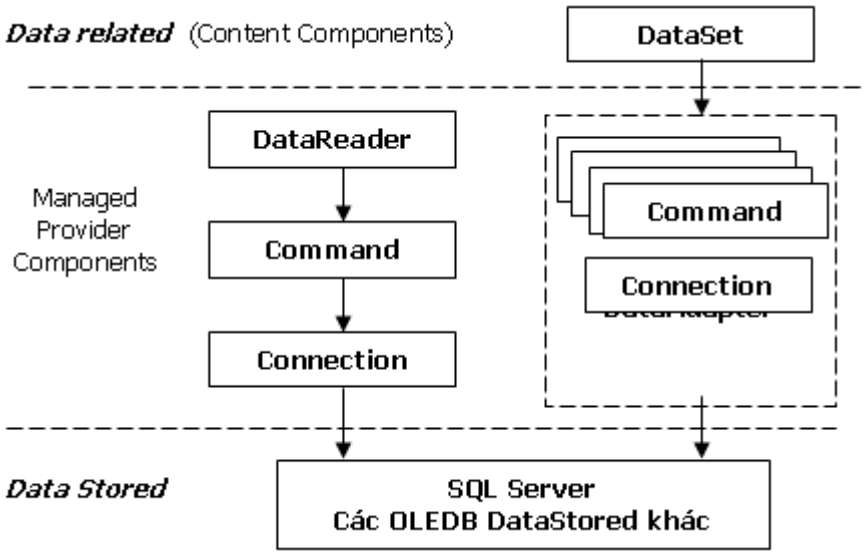
Hầu hết ứng dụng hay các website đều cần phải có cơ sở dữ liệu, để lưu trữ dữ liệu, xử lý thông tin và đưa ra các báo cáo, hỗ trợ tìm kiếm…

Khi dữ liệu trở thành trung tâm của ứng dụng thì cung cấp các chức năng tới người dùng phụ thuộc vào khả năng thao tác dữ liệu, vấn đề mà người thiết kế và người xây dựng ứng dụng quan tâm khi sử dụng dữ liệu là:

- Lưu dữ liệu tập trung
- Đảm bảo toàn vẹn dữ liệu
- Đảm bảo khả năng truy xuất đồng thời của nhiều người dùng trên dữ liệu
- Đảm bảo thời gian hồi đáp ngắn cho mỗi người dùng
- Bảo mật dữ liệu
- Trao đổi dữ liệu giữa các hệ thống khác nhau

Những vấn đề này được giải quyết dựa vào khả năng của các hệ quản trị cơ sở dữ liệu(HQT CSDL) và các phần mềm xử lý dữ liệu do HQL CSDL cung cấp. Net truy xuất dữ liệu qua ADO.NET, đặc điểm chính của ADO.NET là khả năng làm việc vơi dữ liệu không kết nối, dữ liệu được lưu trữ trong bộ nhớ như một csdl thu nhỏ gọi là dataset, nhằm tăng tốc độ tính toán, xử lý tính toán và hạn chế sử dụng tài nguyên trên Database Server. ĐẶc điểm quan trọng thứ 2 là khả năng xử lý dữ liệu chuẩn XML, dữ liệu ở dạng XMl có thể trao đổi giữa bất kỳ hệ thống nào nên ứng dụng của bạn sẽ có nhiều khả năng làm việc với nhiều ứng dụng khác.

## I. Kiến trúc ADO .Net




Hình 1

Kiến trúc ADO.NET có thể chia làm 2 phần chính:

- ***Managed Provider Component: bao gồm các đối tượng như DataAdapter, DataReader,…*** giữ nhiệm vụ làm việc trực tiếp với dữ liệu như database, file,…

- ***Content Component: bao gồm các đối tượng như DataSet, DataTable,…***
  đại diện cho dữ liệu thực sự cần làm việc. DataReader là đối tượng mới, giúp truy cập dữ liệu nhanh chóng nhưng forward-only và read-only giống như ADO RecordSet sử dụng Server cursor, OpenFowardOnly và LockReadOnly. DataSet cũng là một đối tượng mới, không chỉ là dữ liệu, DataSet có thể coi là một bản sao gọn nhẹ của CSDL trong bộ nhớ với nhiều bảng và các mối quan hệ.
  DataAdapter là đối tượng kết nối giữa DataSet và CSDL, nó bao gồm 2 đối tượng Connection và Command để cung cấp dữ liệu cho DataSet cũng như cập nhật dữ liệu từ DataSet xuống CSDL.

Trước khi đi vào học cụ thể các đối tượng của ADO.NET chúng ta cùng xem qua một ví dụ HelloWorld với ADO.NET qua đó bạn sẽ thấy được công việc cần thực hiện khi thao tác với database(ở ví dụ này mình dùng với SQLExpress).

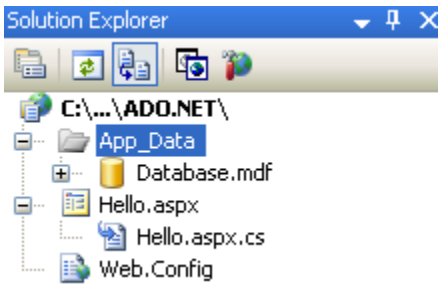
Để làm ví dụ này bạn thực hiện theo các bước sau:

- bước 1. Nhấn chuột phải vào thư mục App_Data chọn new Item, Cửa sổ Add New Item hiện ra bạn chọn SqlDatabase như hình 1 sau

Hình 2

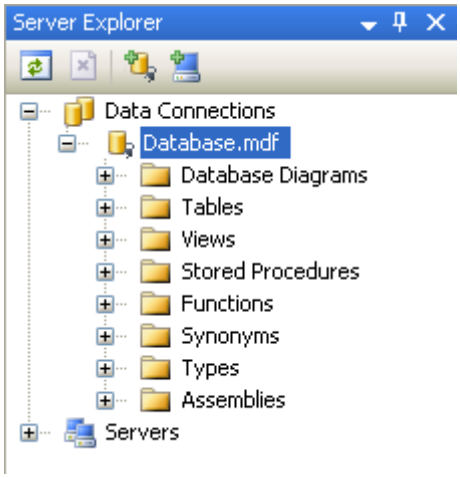
- bước 2. Nhập tên Database vào hộp Name sau đó nhấn Add.

  Trong Solution Explorer sẽ thêm vào Database trong thư mục App_Data.



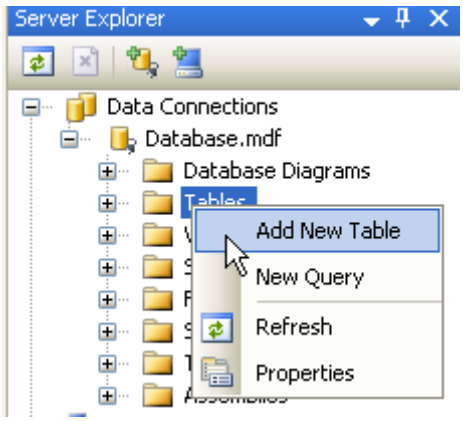

Hình 3

- bước 3. tạo bảng dữ liệu cho Database.mdf
    - bước 3.1 bạn click đúp chuột vào Datatabase.mdf -> Server Explorer hiện ra như sau:



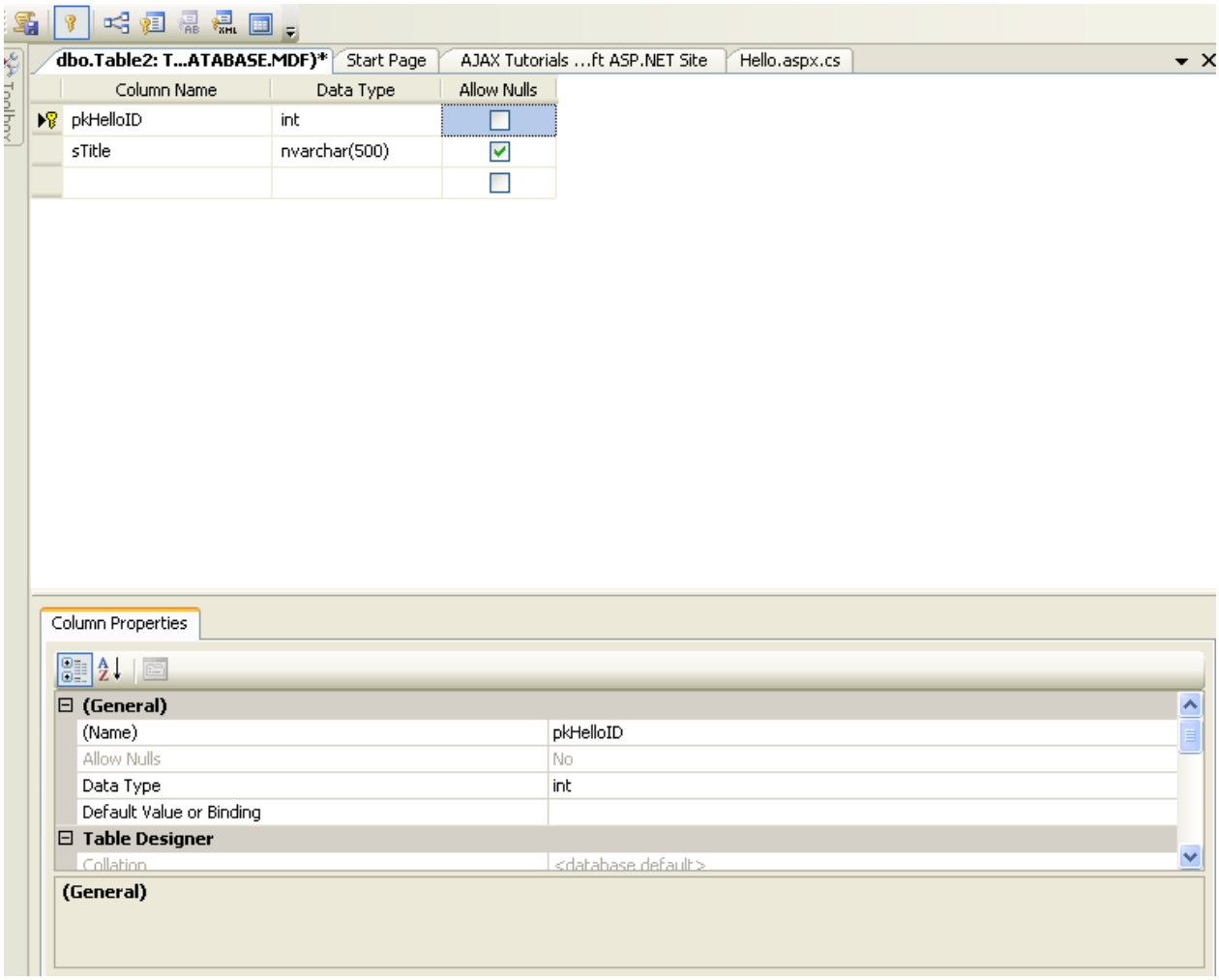

Hình 4

    - bước 3.2 bạn nhấn chuột phải vào Tables và chọn Add New Table




Hình 5

Trong màn hình của VS sẽ hiện ra như hình 6 và bạn thao tác tạo các trường dữ liệu như thao tác với Access hay MSSQL 2000/2005




Hình 6

- bước 4.Viết code cho Hello.aspx.cs

```
using System;
using System.Data;
using System.Data.SqlClient;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        //chuỗi kết nối đến nguồn dữ liệu
        string driver = "Data
Source=(local)\\SQLEXPRESS;AttachDbFilename=|DataDirectory|Database.mdf
;Integrated Security=True;User Instance=True";
        //đối tượng kết nối tới cơ sở dữ liệu
        SqlConnection sqlconn = new SqlConnection(driver);
        //Command điều khiển truy vấn sql
        SqlCommand sqlcom = sqlconn.CreateCommand();
```

```
        sqlcom.CommandText = "select sTitle from tblHello where
pkHelloID=1";
        //mở kết nối dữ liệu
        sqlconn.Open();
        //lấy về chuỗi giá trị trong cơ sở dữ liệu
        string result = (string)sqlcom.ExecuteScalar();
        //đóng kết nối
        sqlconn.Close();
        //in giá trị ra màn hình
        Response.Write(result);
    }
}
```

Cơ bản các bước thực hiện với database

- bước 1: Tại kết nối
- bước 2: Tạo lệnh điều khiển truy vấn SQL
- bước 3:Mở kết nối dữ liệu
- bước 4: thực thi lệnh
- bước 5: đóng kết nối
- bước 6: in kết quả

## II. Đối tượng Connection

Kết nối cơ sở dữ liệu SQLServer

Bạn cần nhập khẩu lớp SqlClient

```
using System.Data.SqlClient;
```

Khai báo và khởi tạo:

```
SqlConnection sqlcon;
string driver="server=localhost; UID=sa; PWD=;
database=name_database";
sqlcon=new Sqlconnection();
sqlcon.ConnectionString=driver;
```

Driver là chuỗi kết nối đến cơ sở dữ liệu trong trường hợp này mình kết nối với sqlserver 2000/2005

Kết nối với cơ sở dữ liệu Access

Bạn cận nhập khẩu lớp OleDb

```
using System.Data.OleDb;
OleDbConnection oleconn;
string driver = "Provider=Microsoft.jet.OLEDB.4.0; Data
Source=duongdan_tendata";
oleconn = new OleDbConnection();
oleconn.ConnectionString = driver;
```

**1. Thuộc tính:**

ConnectString: chứa đựng chuỗi kết nối tới cơ sở dữ liệu

Database: Chứa đựng tên cơ sở dữ liệu trong chuỗi kết nối ConnectString ở trên và bạn có thể thay đổi cơ sở dữ liệu trong lúc thực thi bằng phương thức ChangeDataBase:

Sqlconn.ChangeDatabase("name_database_thaydoi");

Server: tên máy chủ bạn trỏ tới

Connect Timeout: số thời gian(tính bằng giây) chờ kết nối dữ liệu mặc đình là 15giây, nếu trong khoảng thời gian này mà vẫn chưa kết nối xong một lỗi Connect Timeout được đưa ra.

State: trả về trạng thái của đối tượng SqlConnection: bạn có thể kiểm tra trạng thái của State như sau

```
Response.Write(sqlconn.State)
```

**2. Phương thức của đối tượng Connection**

Các phương thức của đối tượng Connection

Open: cho phép mở dữ liệu với các thuộc tính đã khai báo trong ConnectString

Close: Đóng cơ sở dữ liệu đang mở

CreateCommand: phương thức cho phép gán hay trả về một đối tượng Command ứng với đối tượng Connection, như ví dụ HelloWorld

```
SqlConnection sqlconn = new SqlConnection(driver);
SqlCommand sqlcom = sqlconn.CreateCommand();
sqlcom.CommandText = "select sTitle from tblHello where pkHelloID=1";
```

BeginTransaction: Phương thức này khai báo bắt đầu một chuyển tác, để kết thúc chuyển tác bạn dùng Table Commit

Rollback: trong trương hợp có lỗi trong quá trình thực thi bạn có thể sử dụng phương thức Rollback để huỷ bỏ các chuyển tác đã thực hiện.

Dispose: dùng để huỷ bỏ hay giải phóng đối tượng Connection đang sử dụng

## III. Đối tượng SqlCommand

Khai báo và khởi tạo đối tượng

Cách 1:

```
SqlCommand sqlcom;
sqlcom=new SqlCommand(ssql,sqlconn)
```

cách 2:

```
SqlCommand sqlcom = new SqlCommand();
sqlcom.Connection = sqlconn;
sqlcom.CommandType = CommandType.Text;
```

```
        sqlcom.CommandText = "select sTitle from tblHello where
pkHelloID=1";
```

Phương thức

- ExcuteReader: dùng để thực thi đọc cơ sở dữ liệu từ bảng cơ sở dữ liệu

- ExcuteNoneQuery: Dùng để thực thi các phát biểu T-Sql như: Insert, Update, Delete, Create,…

- ExcuteScalar: trả về từ phát biết SQL dạng Select chỉ có một cột một hàng.

## IV. Đối tượng SqlDataReader

đối tượng này được net cung cấp để đọc dữ liệu từ bảng cơ sở dữ liệu, nó là đối tượng chỉ phục vụ thao tác đọc dữ liệu(Read only). Trong khi truy xuất dữ liệu nó sẽ giữ kết nối liên tục với database(hướng kết nối)

Khai báo và khởi tạo đối tượng

```
SqlDataReader sqlreader;
sqlreader = sqlcom.ExecuteReader();
```

## V. Đối tượng DataAdapter

OleDataAdapter được xem như bộ đọc dữ liệu từ cơ sở dliệu nguồn và điền chúng vào đối tượng DataSet hay DataTable

Khai báo, khởi tạo và giải phóng đối tượng.

```
       string ssql;
Khai báo đối tượng
       Dim sqlcom As SqlCommand
       Dim sqlconn As SqlConnection
       Dim sqladapter As SqlDataAdapter
       sqlconn.Open();
```

cách 1.

```
            sqladapter = New OleDbDataAdapter(ssql, sqlconn)

            sqlcom = New SqlCommand(ssql, sqlconn)
cách 2.
            sqladapter = new SqlDataAdapter(sqlcom);
Giải phóng đối tượng
            sqladapter.Dispose();
```

Thuộc tính:

Các thuộc tính bao gồm SelectCommand, InsertCommand, UpdateCommand, DeleteCommand: thực hiện các thao tác select, insert, update, delete dữ liệu

Phương thức:

Fill: Phương thức thực thi câu lệnh select trong sql rồi điền kết quả cho DataSet hoặc Datatable.

Update: gọi lệnh câph nhật các thay đổi vào dữ liệu lên các dữ liệu nguồn

Điền dữ liệu từ Adapter vào DataSet

Dataset là một thùng chứa dữ liệu không kết nối

```
public static DataSet Filldataset(string ssql)
{
    DataSet dataset = new DataSet();
    opendata();
    try
    {
        sqladapter = new SqlDataAdapter(ssql, sqlconn);
        sqladapter.Fill(dataset);
        sqladapter.Dispose();
    }
    catch (Exception exp)
    {
        closedata();

System.Web.HttpContext.Current.Response.Write(exp.ToString());
    }
    closedata();
    return dataset;
}
```

Điền dữ liệu vào DataTable

```
public static DataTable FillDatatable(string ssql)
{
    opendata();
    DataTable datatable = new DataTable();
    try
    {
        sqladapter = new SqlDataAdapter(ssql, sqlconn);
        sqladapter.Fill(datatable);
        sqladapter.Dispose();
    }
    finally
    {
        closedata();
    }
    closedata();
    return datatable;
}
```

## VI. Đối tượng Dataset và DataTable

Là thành phần chính của kiến trúc không kết nối cơ sở dữ liệu, được dùng để nắm giữ dữ liệu của mọi cơ sở dữ liệu và cho phép thay đổi dữ liệu bên trong đối tượng này để sau đó cập nhật trở lại cơ sở dữ liệu nguồn bằng phương thức Update của đối tượng DataAdapter

Khởi tạo

```
DataSet dataset = new DataSet();
DataSet dataset = new DataSet("Mydataset");
```

Thuộc tính Tables, dataset được dùng để chứa danh sách các đối tượng DataTable

Ví dụ:

```
        private void button1_Click(object sender, EventArgs e)
        {
            string strQuery = "select * from tblEmployees";
            DataSet dataSet = new DataSet("Employees");
            try
            {
                SqlDataAdapter sqlDataAdapter = new
                    SqlDataAdapter(strQuery, Connection.sqlConnection);
                sqlDataAdapter.Fill(dataSet);

                sqlDataAdapter.Dispose();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error: " + ex.Message);
            }
            this.dataGridView1.DataSource = dataSet.Tables[0];

            //lay ve ten cua doi tuong dataset
            label1.Text ="DataSetName: " + dataSet.DataSetName ;

        }
        private void button2_Click(object sender, EventArgs e)
        {

//khai bao phat bieu sql 1
            string strQuery = "select * from tblEmployees";
            DataSet dataSet = new DataSet("Employees");
            try
            {
                SqlDataAdapter sqlDataAdapter = new
                    SqlDataAdapter(strQuery, Connection.sqlConnection);
                sqlDataAdapter.Fill(dataSet);

//khai bao phat bieu sql 2
                strQuery = "select * from tblContracts";
                sqlDataAdapter = new
                    SqlDataAdapter(strQuery, Connection.sqlConnection);
                DataTable dataTable = new DataTable();
                sqlDataAdapter.Fill(dataTable);
                dataSet.Tables.Add(dataTable);

                sqlDataAdapter.Dispose();
                string dataTableName="";
                foreach(DataTable dt in dataSet.Tables)
                {
                    dataTableName += dt.TableName + " ";
                }
                label1.Text = "Number of tables: " + dataTableName ;
            }
            catch (Exception ex)
```

```
            {
                MessageBox.Show("Error: " + ex.Message);
            }

            this.dataGridView1.DataSource = dataSet.Tables[1] ;

        }
```

Phuong thuc Add, Remove

DataSet dataset=new DataSet();

DataTable datatable=new DataTable("datatablename");

dataset.Tables.Add(datatable);

dataset.Tables.Remove(datatable);

xoa voi datatable duoc dat tên

dataset.Tables.Remove(datatablename);

dataset.Tables.RemoveAt(0);

phương thức Clear loại bỏ tất cả các đối tượng trong DataTable

dataset.Tables.Clear();

De dem so dong du lieu trong bang ta co the thuc hien

```
int sodong=dataset.Tables[0].Rows.Count;
```

2. Đối tượng DataTable

```
        private void button1_Click(object sender, EventArgs e)
        {
            string strQuery = "select top 10 * from tblEmployees";
//khoi tao doi tuong DataTable
            dataTable = new DataTable("Employees");
            try
            {
                SqlDataAdapter sqlDataAdapter = new
                    SqlDataAdapter(strQuery, Connection.sqlConnection);
//dien du lieu vao datatable
                sqlDataAdapter.Fill(dataTable);


                sqlDataAdapter.Dispose();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error: " + ex.Message);
            }
//gan du lieu va dataGrid voi thuoc tinh DataSource
            this.dataGridView1.DataSource = dataTable;
            label1.Text= dataTable.TableName ;
        }


// thuoc tinh DataRow tra ve cac mau tin dang chua trong doi tuong
DataTable
```

```
        private void button2_Click(object sender, EventArgs e)
        {
            if (dataTable != null)
            {
                string name = "";
                foreach (DataRow dataRow in dataTable.Rows)
                {
                    name += Convert.ToString(dataRow[1]) + "\n";
                }
                label1.Text = name;
            }
        }

//thuoc tinh Columns tra ve tap doi tuong DataColumn bao gom danh sach
cot du lieu cua bang chua trong doi tuong DataTable
        private void button3_Click(object sender, EventArgs e)
        {
            if (dataTable != null)
            {
                string name = "";
                foreach (DataColumn dataColumn in dataTable.Columns)
                {
                    name += Convert.ToString(dataColumn.ColumnName) +
"\n";
                }
                label1.Text = name;
            }
        }
```

Ví dụ chúng ta sẽ ứng dụng 3 dối tượng trên vào việc, cập nhật và hiển thị dữ liệu cho bảng sản phẩm

Bước 1: tạo bảng cơ sở dữ liệu

Ví dụ chúng ta có một bảng dữ liệu tblIntrodure gồm các trường:

| | |
|---|---|
| pkIntrodureID | (int) |
| sTitle | (nvarchar(300) |
| sSummary | (nText) |
| iContent | (nText) |
| iPosition | (int) |

Bước 2: tạo thủ tục StoreProcedure

ta tạo ra 3 thủ tục sql cho bảng giới thiệu của ta như sau

spIntrodure_insert - Thủ tục thêm mới dữ liệu

```
Create PROCEDURE spIntrodure_insert
      @sTitle nvarchar(100),
      @sSummary ntext,
      @sContent ntext,
      @iPosition int
```

```
AS
      insert into tblIntrodure(sTitle, sSummary, sContent, iPosition)
            values(@sTitle, @sSummary, @sContent, @iPosition)
      GO
```

spIntrodure_edit - Thủ tục sửa dữ liệu

```
Create PROCEDURE spIntrodure_edit
      @pkIntrodureID int,
      @sTitle nvarchar(100),
      @sSummary ntext,
      @sContent ntext,
      @iPosition int
AS
      update tblIntrodure set
            sTitle=@sTitle, sSummary=@sSummary, sContent=@sContent,
iPosition=@iPosition
      where pkIntrodureID=@pkIntrodureID
      GO
```

spIntrodure_deletebyID - Thủ tục xoá dữ liệu

```
Create PROCEDURE spIntrodure_deletebyID
      @pkIntrodureID int
AS
      delete from tblIntrodure where pkIntrodureID=@pkIntrodureID
      GO
```

Chú ý: trên là cách tạo 3 thủ tục theo cú pháp của MSSQL nếu bạn tạo thủ tục SQL trong VS thì từ khoá Create sẽ chuyển thành Alter và GO chuyển thành Return

Bước 3: Tạo các lớp(nằm trong thư mục App_Code)

IntrodureInfo.cs

```
using System;

namespace iTechPro.Modules.Introdure
{
    public class IntrodureInfo
    {

        int _pkIntrodureID;
        public int pkIntrodureID
        {
            get { return _pkIntrodureID; }
            set { _pkIntrodureID = value; }
        }

        string _sTitle;
        public string sTitle
        {
            get { return _sTitle; }
            set { _sTitle = value; }
        }

        string _sImage;
```

```
        public string sImage
        {
            get { return _sImage; }
            set { _sImage = value; }
        }

        string _sSumary;
        public string sSumary
        {
            get { return _sSumary; }
            set { _sSumary = value; }
        }

        string _sComment;
        public string sComment
        {
            get { return _sComment; }
            set { _sComment = value; }
        }

        int _iPosition;
        public int iPosition
        {
            get { return _iPosition; }
            set { _iPosition = value; }
        }
    }

}
```

IntrodureDB.cs (chứa tất cả phương thức xử lý và lấy dữ liệu cho bảng tblIntrodure)

```
using System;
using System.Data;
using System.Data.SqlClient;

using Website.Library;

namespace Website.Modules.Introdure
{
    public class IntrodureDB : ExcuteDataHelper
    {

        public IntrodureDB()
        {
            //
            // TODO: Add constructor logic here
            //
        }

        public static void Delete(string _pkIntrodureID)
        {
            string[] parameters = new string[] { "@pkIntrodureID"};
            string[] values = new string[] { _pkIntrodureID};
            executeData("spIntrodure_deletebyID", parameters, values);
```

```
        }

        public static void Insert(IntrodureInfo _introdure)
        {
            string[] parameters = new string[7] { "@sTitle", "@sImage",
"@sSumary", "@sComment", "@sPage", "@sLang", "@iPosition" };
            string[] values = new string[7] { _introdure.sTitle,
_introdure.sImage, _introdure.sSumary, _introdure.sComment,
_introdure.sPage, _introdure.sLang, _introdure.iPosition.ToString() };
            executeData("spIntrodure_insert", parameters, values);
        }

        public static void Update(IntrodureInfo _introdure)
        {
            string[] parameters = new string[7]
{ "@pkIntrodureID" ,"@sTitle", "@sImage", "@sSumary", "@sComment",
"@sPage", "@iPosition" };
            string[] values = new string[7]
{ _introdure.pkIntrodureID.ToString(), _introdure.sTitle,
_introdure.sImage, _introdure.sSumary, _introdure.sComment,
_introdure.sPage, _introdure.iPosition.ToString() };
            executeData("spIntrodure_edit", parameters, values);
        }

        public static void UpdateIndex(string _pkIntrodureID, string
_giatri)
        {
            string ssql = "update tblIntrodure set iPosition=" +
_giatri + " where pkIntrodureID=" + _pkIntrodureID;
            executeData(ssql);
        }

        public static IntrodureInfo Getinfo(string _pkIntrodureID)
        {
            DataTable mydata =
iTechProData.FillDatatable("spIntrodure_selectbyID", "@pkIntrodureID",
_pkIntrodureID);
            IntrodureInfo _introdure = new IntrodureInfo(); ;
            _introdure.sTitle = mydata.Rows[0]["sTitle"].ToString();
            _introdure.sImage = mydata.Rows[0]["sImage"].ToString();
            _introdure.sSumary = mydata.Rows[0]["sSumary"].ToString();
            _introdure.sComment =
mydata.Rows[0]["sComment"].ToString();
            _introdure.sPage = mydata.Rows[0]["sPage"].ToString();
            _introdure.sLang = mydata.Rows[0]["sLang"].ToString();
            _introdure.iPosition =
int.Parse(mydata.Rows[0]["iPosition"].ToString());
            return _introdure;
        }
    }

}
```

Tại lớp IntrodureDB này chúng ta sẽ kế thừa các phương thức thực thi dữ liệu từ lớp ExcuteDataHelper.cs

Lớp ExcuteDataHelper.cs

```
using System;
using System.Data;
using System.Data.SqlClient;


namespace Website.Library
{
    public class ExcuteDataHelper : iTechProData
    {
        //phuong thuc thuc thi du lieu(them moi, chinh sua, xoa) khi
dua vao mot tham so sql
        #region executeData(string sql)"Thực thi dữ liệu"
        public static void executeData(string sql)
        {
            opendata();
            sqlcom = new SqlCommand(sql, sqlconn);
            try
            {
                sqlcom.ExecuteNonQuery();
                closedata();
            }
            catch (Exception exp)
            {
                closedata();
                HttpContext.Current.Response.Write(sql + "<br/>");
                HttpContext.Current.Response.Write("Có lỗi trong quá
trình thực thi " + exp.ToString());
            }
        }
        #endregion

        //phuong thuc thuc thi du lieu voi tham so dua vao
        #region executeData(string store, string[] Parameter, string[]
Values)
        public static void executeData(string store, string[] Parameter,
string[] Values)
        {
            opendata();
            sqlcom = new SqlCommand();
            sqlcom.CommandText = store;
            sqlcom.Connection = sqlconn;
            sqlcom.CommandType = CommandType.StoredProcedure;
            for (int i = 0; i < Parameter.Length; i++)
            {
                sqlcom.Parameters.AddWithValue(Parameter[i],
Values[i]);
            }
            try
            {
                sqlcom.ExecuteNonQuery();
                closedata();
            }
            catch (DataException exp)
            {
                sqlconn.Close();
                HttpContext.Current.Response.Write(exp.ToString());
            }
```

```
        }
        #endregion
    }
}
```

Trong lớp này chúng ta có 2 phương thức thực thi dữ liệu có thể là thêm mới, chỉnh sửa hay xoá dữ liệu void executeData(string sql) cho phép bạn thực thi dữ liệu với một chuỗi sql đưa vào còn executeData(string store, string[] Parameter, string[] Values) sẽ thực thi dữ liệu với hàm thủ tục từ SQL truyền vào với hai mảng giá trị và tham số và lop này này thừa kế từ lớp dẫn xuất iTechProData.cs

Lớp iTechProData.cs

```
using System;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;


namespace Website.Library
{
    public class WebsiteData
    {
        #region khai bao bien

        protected string ssql;
        protected static SqlConnection sqlconn;
        protected static SqlCommand sqlcom;
        protected static SqlDataAdapter sqladapter;
        protected static DataSet mydata;
        protected static SqlDataReader sqlreader;

        #endregion

        //phuong thuc mo du lieu
        #region opendata() "Mở dữ liệu"
        public static void opendata()
        {
            //đọc chuỗi kết nối từ trong file web.config
            System.Configuration.AppSettingsReader settingsReader = new
AppSettingsReader();
            string driver =
(string)settingsReader.GetValue("hcubiudata", typeof(String));

            try
            {
                sqlconn = new SqlConnection(driver);
                if (sqlconn.State != ConnectionState.Open)
                {
                    sqlconn.Open();
                }
            }
            catch (Exception exp)
            {
                HttpContext.Current.Response.Write("Lỗi mở dữ liệu" +
exp.ToString());
```

```
            }
        }
        #endregion

        //phuong thuc dong du lieu
        #region closedata() "Đóng dữ liệu"
        public static void closedata()
        {
            if (sqlconn.State != ConnectionState.Closed)
            {
                sqlconn.Close();
                sqlconn.Dispose();
            }
        }
        #endregion

        // điền dữ liệu vào DataTable từ một thủ tục trong Database
        public static DataTable FillDatatable(string store,string
_thamso, string _giatri)
        {
            opendata();
            DataTable datatable = new DataTable();
            sqlcom = new SqlCommand();
            sqlcom.CommandText = store;
            sqlcom.Connection = sqlconn;
            sqlcom.Parameters.AddWithValue(_thamso, _giatri);
            sqlcom.CommandType = CommandType.StoredProcedure;
            try
            {
                sqladapter = new SqlDataAdapter(sqlcom);
                sqladapter.Fill(datatable);
                sqladapter.Dispose();
            }
            finally
            {
                closedata();
            }
            return datatable;
        }
    }

}
```

Trong lớp trên bạn thấy có 2 đối tượng data mới đó là DataAdapter và DataTable chúng ta sẽ học kỹ hơn trong phần sau trong ví dụ này các bạn chỉ cần hiểu qua là DataAdapter là bộ đọc dữ liệu từ nguồn dữ liệu, và DataTable là đối tượng lưu trữ dữ liệu không kết nối, nó như một bảng tạm để chứa dữ liệu và nó ko cần biết dữ liệu đó từ nguồn nào.

Bước 4: Tạo giao diện sử dụng

Code: adminIntrodure.aspx

```
<%@ Page Language="C#" MasterPageFile="~/admin.master"
AutoEventWireup="true" CodeFile="adminIntrodure.aspx.cs"
Inherits="Desktop_Introdure_adminIntrodure" Title="Admin -
Introdure" %>
```

```
<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">
<!--Trinh bay du lieu-->
 <table cellpadding="0" cellspacing="0" width="100%" style="padding-
right:3px; height:390px">

  <tr>
   <td style="padding:15px 15px 15px 15px" valign="top">

<table width="100%" cellpadding="0" cellspacing="0">
    <tr>
     <td align="left" class="hcubiufontlarger">Giới thiệu</td>
    </tr>
    <tr><td style="height:15px;"></td></tr>


   <!--start them moi-->
   <tr>
    <td align="left">
        <asp:Panel ID="panelupdate" Width="100%" runat="Server"
Visible="false">
            <table width="100%" style="padding-left:20px;">
             <tr>
              <td colspan="2"><b>Cập nhật thông tin giới thiệu</b></td>
             </tr>
             <tr><td style="width: 78px; height:15px;"></td></tr>
             <tr>
                <td align="left" style="width: 78px">Tiêu đề</td>
                <td align="left"><input type="text" name="txtTitle"
id="txtTitle" runat="server" style="width: 329px" /></td>
             </tr>
             <tr>
                <td valign="middle" align="left" style="width:
78px">Tóm tắt</td>
                <td align="left">

                    <asp:TextBox ID="txtTomtat" runat="server"
TextMode="MultiLine"></asp:TextBox>

                </td>
             </tr>
             <tr>
                <td align="left" style="height: 88px; width: 78px;">Nội
dung</td>
                <td align="left" style="height: 88px">

                  <asp:TextBox ID="txtNoidung" runat="server"
TextMode="MultiLine" />

                </td>
             </tr>
             <tr>
              <td align="left">Vị trí</td>
              <td align="left">
```

```
                <asp:TextBox ID="txtvitri" runat="server"
Text="1"></asp:TextBox>
                  <asp:RangeValidator ID="RangeValidator1"
runat="server" ControlToValidate="txtvitri"
                      ErrorMessage="Vị trí phải là kiểu số"
MaximumValue="100" MinimumValue="0"
Type="Integer"></asp:RangeValidator></td>
              </tr>
              <tr><td style="width: 78px; height:15px;"></td></tr>
              <tr>
               <td colspan="2" align="left">
                <asp:Button ID="btnaccept" runat="server" Text="Ghi"
Width="100px" OnClick="btnaccept_Click"/>
                 <asp:Button ID="btcancel" runat="server" Text="Bỏ qua"
Width="100px" OnClick="btcancel_Click" />
                  <asp:Label ID="lblidintro" runat="server" Text=""
Visible="false"></asp:Label></td>
              </tr>
            </table>
          </asp:Panel>

<!--End them moi-->

    </td>
   </tr>

   <tr>
    <td  style="height:5px;"></td>
   </tr>

   <asp:Panel ID="panelview" runat="server">

   <tr>
    <td align="left" style="padding-bottom:3px;"><asp:LinkButton
ID="btnaddnew" CssClass="linkbutton" runat="server" Text="Thêm mới"
OnClick="btnaddnew_Click" /></td>
   </tr>

   <tr>
    <td valign="top" align="left">
    <asp:DataGrid id="gridintro" runat="server"
        BorderColor="black"
        Width="100%"
        BorderWidth="1"
        CellPadding="3"
        Font-Size="10pt"
        HeaderStyle-BackColor="#aaaadd"
        OnItemCommand="gridintro_OnItemCommand"
        AutoGenerateColumns="false">
        <HeaderStyle BackColor="#AAAADD"></HeaderStyle>
        <Columns>
          <asp:TemplateColumn HeaderStyle-HorizontalAlign="Center"
ItemStyle-HorizontalAlign="Center" HeaderStyle-Width="80px"
HeaderText="STT">
            <ItemTemplate>
              <%#Container.ItemIndex +1 %>
            </ItemTemplate>
```

```
          </asp:TemplateColumn>
          <asp:BoundColumn HeaderStyle-HorizontalAlign="Left"
ItemStyle-HorizontalAlign="Left" DataField="sTitle" ReadOnly="true"
HeaderText="Tiêu đề"></asp:BoundColumn>

          <asp:TemplateColumn HeaderText="Vị trí" ItemStyle-
HorizontalAlign="Center" HeaderStyle-HorizontalAlign="Center"
HeaderStyle-Width="100px" ItemStyle-Width="100px" ItemStyle-
Height="24px" >
           <ItemTemplate>
            <asp:TextBox ID="txtVitri" Width="39px" runat="server"
Text='<%#Eval("iPosition") %>' />
           </ItemTemplate>
          </asp:TemplateColumn>

          <asp:TemplateColumn HeaderText="Chỉnh sửa" ItemStyle-
HorizontalAlign="Center" HeaderStyle-Width="80px" HeaderStyle-
HorizontalAlign="Center" ItemStyle-Width="100px" ItemStyle-
Height="24px" >
           <ItemTemplate>
            <asp:LinkButton ID ="Edit" CommandArgument
='<%#DataBinder.Eval(Container,"DataItem.pkIntrodureID")%>' runat
="server" CommandName="Edit" Text ="Edit"></asp:LinkButton>
           </ItemTemplate>
          </asp:TemplateColumn>

          <asp:TemplateColumn HeaderText="Xóa" HeaderStyle-
HorizontalAlign="Center" HeaderStyle-Width="80px" ItemStyle-
HorizontalAlign="Center" ItemStyle-Width="100px" ItemStyle-
Height="24px" >
           <ItemTemplate>
            <asp:LinkButton ID ="Delete" CommandArgument
='<%#DataBinder.Eval(Container,"DataItem.pkIntrodureID")%>' runat
="server" CommandName="Delete" Text ="Delete"></asp:LinkButton>
           </ItemTemplate>
          </asp:TemplateColumn>

         </Columns>
     </asp:DataGrid>
    </td>
  </tr>
  <tr>
   <td align="right" style="padding-top:3px;">
    <asp:Label ID="lblthongbao" runat="server"></asp:Label>
    <asp:LinkButton ID="lbncapnhatvitri" CssClass="linkbutton"
runat="server" Text="Cập nhật vị trí" OnClick="lbncapnhatvitri_Click"
/>
   </td>
  </tr>

  </asp:Panel>
 </table>

   </td>
  </tr>
  <tr><td style="height:30px;"></td></tr>
 </table>
```

```
</asp:Content>
```

## Code adminIntrodure.aspx.cs

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
using Website.Library;
using Website.Modules.Introdure;

public partial class Desktop_Introdure_adminIntrodure :
System.Web.UI.Page
{
    string ssql;
    void Loaddatagrid()
    {
        ssql = "select pkIntrodureID,sTitle,iPosition from
tblIntrodure";
        DatagridHelper.fill_datagrid(gridintro, ssql, "pkIntrodureID");
        foreach (DataGridItem item in this.gridintro.Items)
        {
            LinkButton lbn =
(LinkButton)this.gridintro.Items[item.ItemIndex].FindControl("Delete");
            lbn.Attributes.Add("onclick", "javascript:return
confirm('Bạn có chắc chắn xoá mục giới thiệu này')");
        }
    }

    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            Loaddatagrid();
        }
    }

    private IntrodureInfo Getcontent()
    {
        IntrodureInfo intro = new IntrodureInfo();
        try
        {
            intro.pkIntrodureID = int.Parse(lblidintro.Text);
        }
        catch
        {
        }
        intro.sTitle = txtTitle.Value;
        intro.sSumary = txtTomtat.Text;
        intro.sContent = txtNoidung.Text;
```

```
        intro.iPosition = int.Parse(txtvitri.Text);
        return intro;
    }

    protected void btnaddnew_Click(object sender, EventArgs e)
    {
        panelupdate.Visible = true;
        panelview.Visible = false;
        txtNoidung.Text = "";
        txtTitle.Value = "";
        this.txtTomtat.Text = "";
        txtvitri.Text = "1";
        btnaccept.Text = "Ghi";
    }

    protected void gridintro_OnItemCommand(object sender,
DataGridCommandEventArgs e)
    {
        lblidintro.Text = e.CommandArgument.ToString();
        if (e.CommandName == "Edit")
        {
            IntrodureInfo introdure =
IntrodureDB.Getinfo(lblidintro.Text);
            txtTitle.Value = introdure.sTitle;
            txtTomtat.Text = introdure.sSumary;
            txtvitri.Text = introdure.iPosition.ToString();
            txtNoidung.Text = introdure.sContent;
            btnaccept.Text = "Cập nhật";
            panelupdate.Visible = true;
            panelview.Visible = false;
        }
        else
        {
            IntrodureDB.Delete(lblidintro.Text);
            Loaddatagrid();
        }
    }

    protected void btnaccept_Click(object sender, EventArgs e)
    {
        IntrodureInfo introdure = Getcontent();
        if (btnaccept.Text == "Ghi")
        {
            IntrodureDB.Insert(introdure);
        }
        else
        {
            IntrodureDB.Update(introdure);
        }
        panelupdate.Visible = false;
        panelview.Visible = true;
        Loaddatagrid();

      }

    protected void btcancel_Click(object sender, EventArgs e)
    {
```

```
        panelview.Visible = true;
        panelupdate.Visible = false;
        Loaddatagrid();
    }

    protected void lbncapnhatvitri_Click(object sender, EventArgs e)
    {
        foreach (DataGridItem item in gridintro.Items)
        {
            TextBox txt =
(TextBox)this.gridintro.Items[item.ItemIndex].FindControl("txtVitri");

IntrodureDB.UpdateIndex(gridintro.DataKeys[item.ItemIndex].ToString(),
txt.Text);
        }
    }
}
```

Trong đoạn mã trên có sử dụng DataGrid bạn sẽ được học nó kỹ hơn trong phần sau, bây giờ bạn cứ coi nó như là một công cụ để hiển thị dữ liệu.

# Chương 9. Sử dụng ListControl

Trong chương này các bạn sẽ được học các điều khiển trình bày danh sách như DropDownList, RadioButtonList… và kết thúc chương các bạn sẽ được học 1 cách chi tiết để sử dụng các List Control này tạo một Module bình chọn cho trang web của bạn

Điểm chung cho tất cả các điều khiển danh sách là nó gồm 3 thuộc tính chính

Bạn có thể đưa dữ liệu vào DropDownList từ một mảng danh sách hoặc dữ liệu từ một cơ sở dữ liệu:

Thuộc tính quan trọng

DataSource: chỉ đến nguồn dữ liệu

DataTextField: trường dữ liệu được hiển thị

DataValueField: trường dữ liệu thiết lập giá trị với tương ứng với Text hiển thị

Phương thức `OnSelectedIndexChanged`

Xảy ra khi người dùng thay đổi lựa chọn phần tử trên DropDownList

## I. Điều khiển DropdownList

Cho phép hiển thị một danh sách các lựa chọn, người sử dụng chỉ chọn một lựa chọn 1 lần

Ví dụ:

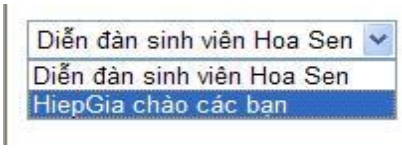
Bạn tạo một lớp phục vụ đưa dữ liệu vào DropDownList như sau:

để sử dụng lớp này bạn tạo 1 trang aspx và trong phần code behind bạn nhập khẩu gói Website.Library, trong trong sự kiện Load của trang bạn gọi như sau

Code 9.1

```
    protected void Page_Load(object sender, EventArgs e)
    {
        DropdownListHelper.Fillcombobox(DropDownList1, "tblIntrodure",
"sTitle", "pkIntrodureID");
    }
```

Kết quả của chương trình sẽ như sau:




Hình 1

Để sử dụng sự kiện OnSelectedIndexchanged bạn cần thêm vào cho DropDownList thuộc tính AutoPostBack và thiết lập cho nó giá trị là true

Code chi tiết

Trang dropdownlist.aspx

Code 9.2

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="dropdownlist.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>DropDownList</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:DropDownList AutoPostBack="true"
OnSelectedIndexChanged="DropDownList1_Changed" ID="DropDownList1"
runat="server">
        </asp:DropDownList><hr />
        <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Trang dropdownlist.aspx.cs

Code 9.3

```
using System;
using Website.Library;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            DropdownListHelper.Fillcombobox(DropDownList1,
"tblIntrodure", "sTitle", "pkIntrodureID");
        }
    }

    protected void DropDownList1_Changed(object sender, EventArgs e)
    {
        Label1.Text = "Text:" + DropDownList1.SelectedItem.Text +
"<br>giá tri:" + DropDownList1.SelectedValue.ToString();
    }
}
```

## II. Sử dụng điều khiển RadiobuttonList

Điều khiển RadioButtonList cho phép hiển thị một danh sách các RadioButton mà có thể sắp xếp theo hướng ngang hay dọc, để ngươi sử dụng có thể chọn một trong các Radiobutton đó.

Ví dụ: khi chúng ta cần thăm dò ý kiến khách hàng về một vấn đề gì đó chúng ta cần tạo một module bình chọn cho website của chúng ta.

Chúng ta sẽ thiết lập 1 bảng sau

bảng tblSurveyAnswer

pkAnswerID (int)

sContent (nvarchar(100))

iVote (int)

iPosition (int)

chúng ta sẽ tạo một trang radiobuttonlist.aspx

Code 9.4

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>RadioButtonList</title>
    <style type="Text/css">
        body{background-color:#e5e5e5}
        #navcontain{width:399px;Height:299px;Background-
color:white;margin:0px auto; padding:15px 15px 15px 15px;}
        A:link{COLOR: #31659C; TEXT-DECORATION: none;}
        A:visited{COLOR: #31659C; TEXT-DECORATION: none;}
        A:active{COLOR: #FC8800; TEXT-DECORATION: none;}
        A:hover{COLOR: #FC8800; TEXT-DECORATION: none;}
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
       <b>Bạn biết đến Điễn đàn Sinh viên Hoa Sen qua:</b><br />
        <asp:RadioButtonList ID="RadioButtonList1" runat="server">
        </asp:RadioButtonList><hr />
       <asp:LinkButton ID="lbnVote" OnClick="lbnVote_Click" Text="Vote"
runat="server" />
       <hr />
       Bạn chọn: <asp:Label runat="server" ID="lblResult" />
    </div>
    </form>
</body>
```

```
</html>
```

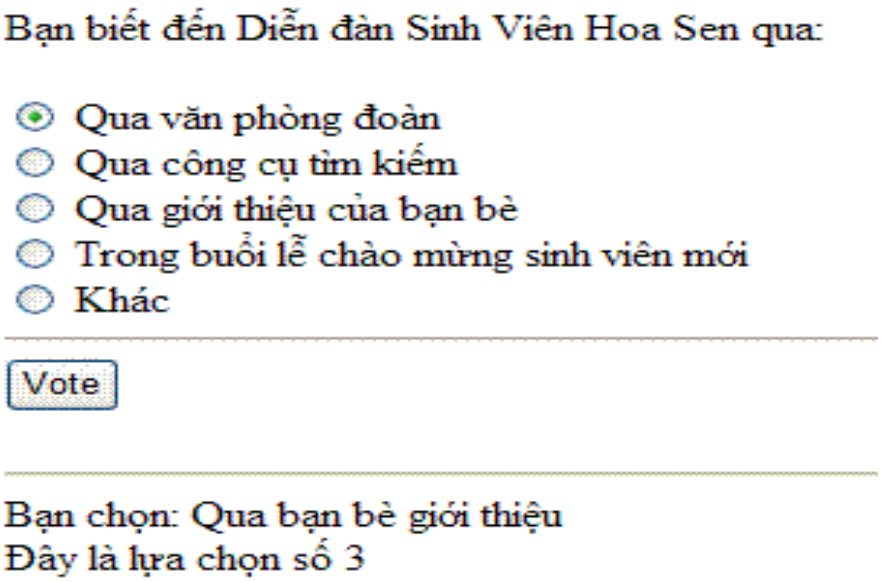
Trang radiobuttonlist.aspx.cs

Code 9.5

```
using System;
using Website.Library;

public partial class radiobuttonlist : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            ListControlHelper.fillRadioButtonList(RadioButtonList1,
"tblSurveyAnswer", "sContent", "pkAnswerID");
        }
    }

    protected void lbnVote_Click(object sender, EventArgs e)
    {
        this.lblResult.Text = RadioButtonList1.SelectedItem.Text +
"<br> và giá trị của nó là:" + RadioButtonList1.SelectedItem.Value;
    }
}
```

Bạn thấy ở Code 9.6 lớp radiobuttonlist.aspx.cs cớ nhập khẩu gói Website.Library có sử dụng phương thức fillRadioButtonList từ lớp ListControlHelper với 4 đối số tương ứng như code ở cuối chương trình

Kết xuất của chương trình




Hình 3

## III. Sử dụng điều khiển ListBox

Nó là một điều khiển giống với DropDownList nhưng nó sẽ hiển thị một danh sách trên trang và chúng ta có thể lựa chọn nhiều phần tử một lúc với thuộc tính selectionMode với hai giá trị là Singer và Multiple.

Ví dụ sau mình sẽ đưa ra với một ListBox nhiều lựa chọn

Code 9.6 ListBox.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        Điều khiển ListBox<br />
        <asp:ListBox SelectionMode="Multiple" ID="listbox1"
runat="server" /><br />
        <asp:Button ID="btnChon" runat="server" Text="Chọn"
OnClick="btnChon_Click" /><hr />
        Bạn đã chọn: <asp:Label ID="lblresult" runat="server" />
    </div>
    </form>
</body>
</html>
```

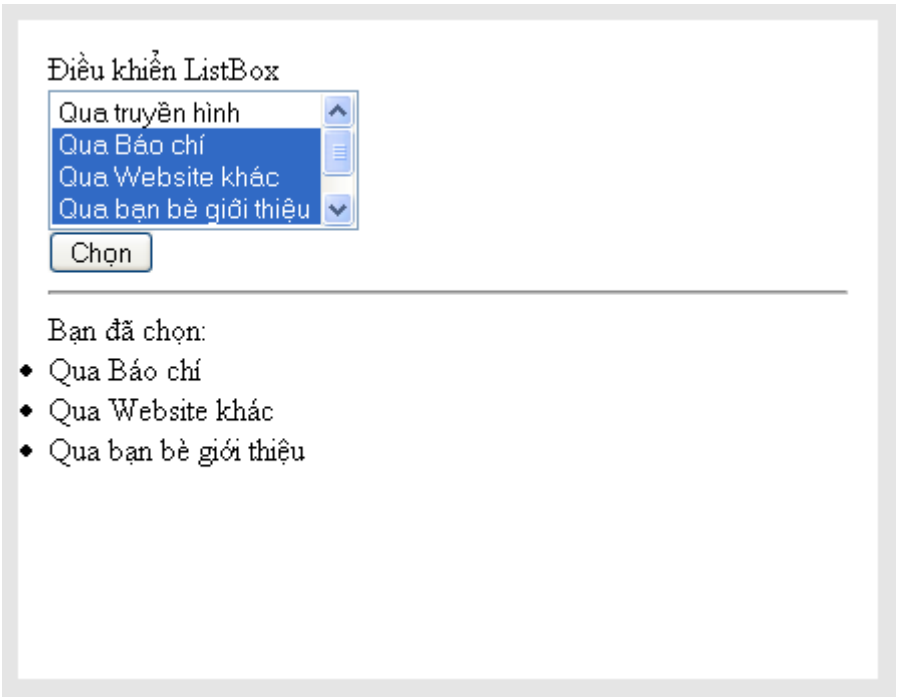
Code 9.7 ListBox.aspx.cs

```
using System;
using System.Collections;
using System.Web;
using System.Web.UI.WebControls;
using Website.Library;

public partial class ListBox : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if(!IsPostBack)
            ListControlHelper.fillListBox(listbox1, "tblSurveyAnswer",
"sContent", "pkAnswerID");
    }

    protected void btnChon_Click(object sender, EventArgs e)
    {
        lblresult.Text = "";
        foreach (ListItem item in listbox1.Items)
        {
```

```
            if (item.Selected)
                lblresult.Text += "<li>" + item.Text;
        }
    }
}
```

Trong code 9.8 ta có sử dụng một hàm fillListBox để điền dữ liệu vào ListBox bạn xem code ở cuối chương.

Kết xuất của chương trình




Hình 4

## IV. Sử dụng điều khiển CheckBoxList

Giống với điều khiển RadioButtonList nhưng nó cho phép người sử dụng chọn lựa nhiều phần tử.

Code 9.9

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>CheckBoxList</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <b>Bạn biết đến Diễn đàn sinh viên Hoa Sen qua</b><br />
```

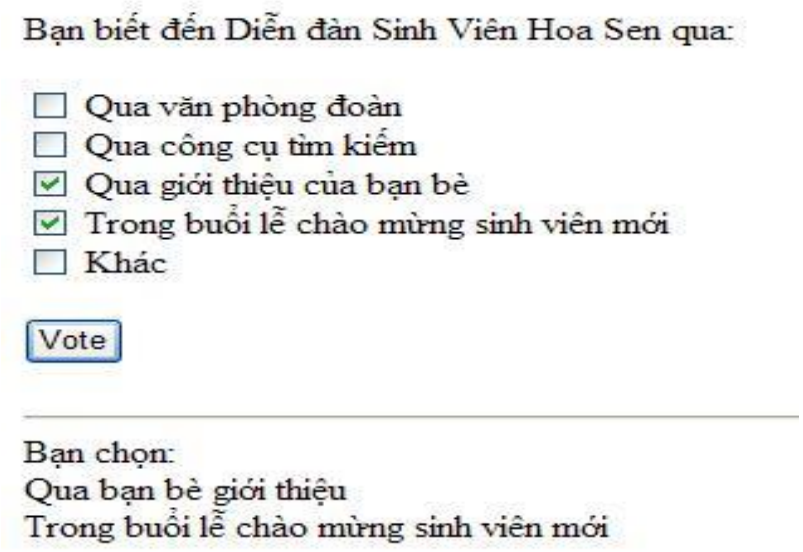
```
        <asp:CheckBoxList ID="CheckBoxList1" runat="server">
        </asp:CheckBoxList><br />
        <asp:Button ID="btnVote" OnClick="btnVote_Click" runat="server"
Text=" Vote " /><hr />
        Bạn đã chọn: <asp:Label ID="lblresult" runat="server" />
    </div>
    </form>
</body>
</html>
```

Code 9.10

```
using System;
using System.Collections;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Website.Library;

public partial class checkBoxList : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            ListControlHelper.fillCheckBoxList(CheckBoxList1,
"tblSurveyAnswer", "sContent", "pkAnswerID");
        }
    }

    protected void btnVote_Click(object sender, EventArgs e)
    {
        lblresult.Text = "";
        for(int i = 0 ;i<CheckBoxList1.Items.Count;i++)
        {
            if (CheckBoxList1.Items[i].Selected == true)
            {
                lblresult.Text += "<li>" + CheckBoxList1.Items[i].Text
+ "</li>";
            }
        }
    }
}
```

Trong code 9.10 có sử dụng hàm fillCheckBoxList là phương thức của lớp ListControlHelper(xem cuối chương) để đưa dữ liệu vào CheckBoxList, trong hàm btnVote_Click được thực hiện khi bạn nhấn vào nút Vote trên trang, phương thức này sẽ duyệt từ Item đầu đến hết trong CheckBoxList và kiểm tra nếu Item đó được chọn thì chúng ta sẽ lấy giá trị

Kết xuất của chương trình




## V. Sử dụng điều khiển BulletedList

Điều khiển này cho phép bạn hiển thị ra kiểu danh sách hay liệt kê, mỗi phân tử của nó có thể đưa ra là Text, linkButton hay một đường dẫn tới một trang web khác

Ví dụ: cũng với bảng dữ liệu trên bạn muốn liệt kê tất cả câu hỏi ra

Code 9.11

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>BulletList Control</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:BulletedList ID="BulletedList1" runat="server">
        </asp:BulletedList>
    </div>
    </form>
</body>
</html>
```

Code 9.12

```
using System;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Website.Library;
```

```
public partial class BulletListItem : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            ListControlHelper.fillBulletList(BulletedList1,
"tblSurveyAnswer", "sContent", "pkAnswerID");
        }
    }
}
```

Kết xuất của chương trình

- Qua truyền hình
- Qua Báo chí
- Qua Website khác
- Qua bạn bè giới thiệu
- Khác

Bạn có thể điều chỉnh sự xuất hiện của bullet trong BulletList với thuộc tính BulletStyle với các giá trị có thể có sau:

Circle, CustomImage, Disc, LowerAlpha, LowerRoman, NotSet, Numbered, Square, UpperAlpha, UpperRoman,

Với thuộc tính có giá trị là CustomImage bạn cần chỉ đến đường dẫn của ảnh trong thuộc tính BulletImageURL

Ví dụ trong Code 9.11 bạn thêm vào thuộc tính Bulletstyle với giá trị là Circle bạn sẽ thấy kết xuất của chương trình như sau:

- Qua truyền hình
- Qua Báo chí
- Qua Website khác
- Qua bạn bè giới thiệu
- Khác

# Chương 10. Sử dụng điều khiển GridView

GridView trình bày dữ liệu như thẻ Table của HTML mà mỗi mục dữ liệu như vói thẻ TR

Chúng ta cùng đi vào xây dựng một lớp gridViewHelper giúp việc điền dữ liệu vào gridView trong các ví dụ của chúng ta.

Trong chương này ngoài điều khiển  ngoài điều khiển GridView các bạn sẽ được giới thiệu thêm về điều khiển sqlDatasource.

Ta đi vào một ví dụ đơn giản: Bạn hiển thị dữ liệu từ bảng Giới thiệu ra 1 GridView

Trong file web.config: bạn thêm vào

```
  <connectionStrings>
    <add name="Gridview" connectionString="Data Source=.\SQLEXPRESS;
AttachDbFilename=|DataDirectory|Database.mdf;Integrated
Security=True;user Instance=True" />
  </connectionStrings>
```

Bạn tạo một trang SimpleGridview.aspx và đưa vào một điều khiển SqlDataSource và điền vào nó các thuộc tính như sau:

Code 10.1

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>GridView</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:GridView AllowSorting="true" DataSourceID="SqlDataSource1"
         ID="GridView1" runat="server">
        </asp:GridView>
        <asp:SqlDataSource
ConnectionString="<%$ ConnectionStrings:Gridview %>"
         SelectCommand="select * from tblIntrodure" ID="SqlDataSource1"
runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Như bạn thấy trong ví dụ trên đối tượng SqlDatasource chứa chỗi kết nối String được lấy ra từ file web.config và thuộc tính selectCommand sẽ đưa vào một chuỗi sql dạng select để lấy tất cả dữ liệu trong bảng tblIntrodure

Và điều khiển GridView của ta sẽ điền vào thuộc tính DataSourceID=”_tên_sqlDatasource”.

Và kết xuất của chương trình sẽ như sau:

| pkIntrodureID | sTitle | sSummary | sContent | iPosition |
|---|---|---|---|---|
| 1 | Tiêu đề diễn đàn ĐH Hoa Sen | Tiêu đề diễn đàn ĐH Hoa Sen | ĐH Hoa Sen | 1 |
| 2 | Hiepgia chào mừng các bạn | Chào mừng các bạn học 1 khoá ASP.NET | Hello | 2 |

Sorting Data

Bạn có thể trình bày sắp xếp dữ liệu trong GridView với thuộc tính AllowSorting

Ví dụ: cũng với ví dụ 1 bạn thêm vào thuộc tính AllowSorting="true" khi này bạn sẽ thấy trên dòng Header của Gridview sẽ xuất hiện như LinkButton và khi bạn nhấn vào nó, nó cho phép bạn sắp xếp thông tin theo thứ tự giảm dần và tăng dần của dữ liệu

Kết xuất của chương trình

| pkIntrodureID | sTitle | sSummary | sContent | iPosition |
|---|---|---|---|---|
| 1 | Tiêu đề diễn đàn ĐH Hoa Sen | Tiêu đề diễn đàn ĐH Hoa Sen | ĐH Hoa Sen | 1 |
| 2 | Hiepgia chào mừng các bạn | Chào mừng các bạn học 1 khoá ASP.NET | Hello | 2 |

Paging Data

Khi số trường dữ liệu lớn bạn có thể thực hiện phân trang cho dữ liệu với việc thiết đặt thuộc tính AllowPaging="true" cũng với ví dụ trên bạn thêm vào thuộc tính AllowPaging, cho nó giá trị bằng true và thiết lập thuộc tính PageSize(số dòng trên một trang) bằng 3 bạn sẽ thấy sự thay đổi

Kết xuất của nó như sau:

| pkIntrodureID | sTitle | sSummary | sContent | iPosition |
|---|---|---|---|---|
| 1 | Tiêu đề diễn đàn ĐH Hoa Sen | Tiêu đề diễn đàn ĐH Hoa Sen | ĐH Hoa Sen | 1 |
| 2 | Hiepgia chào mừng các bạn | Chào mừng các bạn học 1 khoá ASP.NET | Hello | 2 |
| 3 | Phát triển | Phát triển | Phát triển | 3 |

1 | 2

Bạn có thể chỉnh sửa trình bày xuất hiện phân trang theo ý mình thay vì mặc định nó sẽ trình bày bởi những con số ở cuối của GridView với thuộc tính PagerSetting

Ví dụ bạn thêm vào 1 số thuộc tính cho GridView của chúng ta như sau

```
<asp:GridView AllowSorting="true" PageSize="3"
        PagerSettings-Mode="NextPreviousFirstLast" PagerSettings-
Position="TopAndBottom" PagerStyle-HorizontalAlign="Center"
        AllowPaging="true" DataSourceID="SqlDataSource1"
         ID="GridView1" runat="server">
        </asp:GridView>
```

Và bạn thấy kết xuất của nó như sau:

| pkIntrodureID | sTitle | sSummary | sContent | iPosition |
|---|---|---|---|---|
| 1 | Tiêu đề diễn đàn ĐH Hoa Sen | Tiêu đề diễn đàn ĐH Hoa Sen | ĐH Hoa Sen | 1 |
| 2 | Hiepgia chào mừng các bạn | Chào mừng các bạn học 1 khoá ASP.NET | Hello | 2 |
| 3 | Phát triển | Phát triển | Phát triển | 3 |

> | >>

Lớp PagingSetting hỗ trợ các thuộc tính sau:

- FirtPageImageURL: cho phép hiển thị ảnh của liên kết tới trang đầu tiên
- FirstPageText: Cho phép hiển thị Text của liên kết đến trang đầu tiên
- LastPageImageUrl: cho phép hiển thị ảnh của liên kết tới trang cuối cùng.
- LastPageTex: Cho phép hiển thị Text của liên kết đến trang cuối cùng.
- Mode: cho phép bạn lựa chọn hiển thị kiểu cho giao diện phân trang, nó có thể có các giá trị sau:
- NextPrevious, NextPreviousFirstLast, Numeric, and NumericFirstLast.
- NextPageImageUrl: Cho phép hiển thị ảnh liên kết tới trang tiếp theo.
- NextPageText: Text hiển thị cho liên kết đến trang tiếp theo .
- PageButtonCount: hiển thị tổng số trang.
- Position: chỉ định vị trí hiển thị phân trang. Giá trị của nó có thể là: Bottom, Top, and TopAndBottom.
- PreviousPageImageUrl: ảnh hiển thị cho liên kết tới trang trước đó.
- PreviousPageText: Text hiển thị cho liên kết tới trang trước đó.
- Visible: Cho phép hiển thị hay ẩn giao diện phân trang.

Ví dụ tiếp theo chúng ta cùng customize phân trang 1 GridView với PagerTemplate GridView như sau:

Code 10.2 trang GridViewpage.aspx

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

<script runat="server">
    protected void GridView1_DataBound(object sender, EventArgs e)
    {
        Menu menuPager =
(Menu)this.GridView1.BottomPagerRow.FindControl("menuPager");
        for (int i = 0; i < GridView1.PageCount; i++)
        {
            MenuItem item = new MenuItem();
            item.Text = Convert.ToString(i+1);
            item.Value = i.ToString();
            if (GridView1.PageIndex == i)
                item.Selected = true;
            menuPager.Items.Add(item);
            menuPager.DataBind();
        }
    }

    protected void menuPager_MenuItemClick(object sender, MenuEventArgs
e)
    {
        GridView1.PageIndex = Int32.Parse(e.Item.Value);
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Gridview</title>
    <style type="text/css">
        .menu td{padding:5px 0px;}
        .selectedPage a{font-weight:bold;color:red;}
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:GridView ID="GridView1" AllowPaging="true" PageSize="3"
         DataSourceID="SqlDataSource1"
OnDataBound="GridView1_DataBound" runat="server">
         <PagerTemplate>
            <table>
                <tr>
                    <td>
                        <asp:LinkButton id="lnkPrevious" Text="&lt;
Prev" CommandName="Page" CommandArgument="Prev" ToolTip="Previous Page"
Runat="server" />
                    </td>
                    <td>
```

```
                        <asp:Menu id="menuPager"
Orientation="Horizontal" OnMenuItemClick="menuPager_MenuItemClick"
StaticSelectedStyle-CssClass="selectedPage" CssClass="menu"
Runat="server" />
                    </td>
                    <td>
                        <asp:LinkButton id="lnkNext" Text="Next &gt;"
CommandName="Page" CommandArgument="Next" ToolTip="Next Page"
Runat="server" />
                    </td>
                </tr>
            </table>
         </PagerTemplate>
        </asp:GridView>
        <asp:SqlDataSource ID="SqlDataSource1"
ConnectionString="<%$ConnectionStrings:Gridview %>"
         SelectCommand="select * from tblIntrodure"
runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Ở đây trong PagerTemple bạn thêm vào 2 điều khiển Linkbutton và 1 điều khiển Menu để thực hiện phân trang. 2 điều khiển Linkbutton với các thuộc tính Command và CommandArgument được GridView hỗ trợ lên ta không phải viết các phương thức để thực thi còn với điều menu trong sự kiện DataBound của GridView bạn cung cấp một phương thức GridView1_DataBound để điền dữ liệu cho Menu.

Thay đổi dữ liệu trong GridView

Điều khiển GridView chỉ cho phép bạn thay đổi hoặc xoá dữ liệu trong Database được điền vào nó.

Ví dụ sau sẽ hướng dẫn bạn cách chỉnh sửa dữ liệu và xoá dữ liệu trong điều khiển GridView.

Ví dụ trang GridviewEdit.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>GridView</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:GridView AllowSorting="true" PageSize="10"
        PagerSettings-Mode="NextPreviousFirstLast" PagerSettings-
Position="TopAndBottom" PagerStyle-HorizontalAlign="Center"
        AutoGenerateDeleteButton="true"
        AutoGenerateEditButton="true"
        DataKeyNames="pkIntrodureID"
        AllowPaging="true" DataSourceID="SqlDataSource1"
```

```
            ID="GridView1" runat="server">
          </asp:GridView>
          <asp:SqlDataSource
ConnectionString="<%$ ConnectionStrings:Gridview %>"
            SelectCommand="select
pkIntrodureID,sTitle,sSummary,sContent,iPosition from tblIntrodure"
            UpdateCommand="Update tblIntrodure set sTitle=@sTitle,
sSummary=@sSummary, sContent=@sContent,iPosition=@iPosition where
pkIntrodureID=@pkIntrodureID"
            DeleteCommand="Delete from tblIntrodure where
pkIntrodureID=@pkIntrodureID"
            ID="SqlDataSource1" runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Kết xuất của chương trình khi bạn nhấn vào nút "Edit" trên GridView


Khi nhấn vào nút Edit bạn sẽ thấy các TextBox sẽ hiện ra tương ứng với dòng được nhấn và chúng ta có thể thay đổi dữ liệu trong đó để xác nhận thay đổi dữ liệu bạn nhấn Update.

Chú ý rằng GridView sẽ tự động đưa ra CheckBox nếu có trường trong bảng dữ liệu là Boolean. để thay đổi hay xoá dữ liệu bạn phải thiết lập thuộc tính DataKeyNames với giá trị là khoá chính trong bảng cơ sở dữ liệu của bạn.

Hiển thị dữ liệu trống:

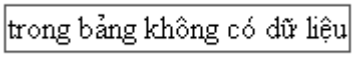
GridView bao gồm hai thuộc tính cho phép bạn hiển thị nội dung cho GridView khi không có dữ liệu, bạn có thể sử dụng EmptyDataText hoặc thuộc tính EmptyDataTemplate để hiển thị nội dung khi dữ liệu rỗng.

Ví dụ trang GridviewdataNull.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>GridView</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:GridView AllowSorting="true" PageSize="10"
        PagerSettings-Mode="NextPreviousFirstLast" PagerSettings-
Position="TopAndBottom" PagerStyle-HorizontalAlign="Center"
        EmptyDataText="trong bảng không có dữ liệu"
        DataKeyNames="pkIntrodureID"
        AllowPaging="true" DataSourceID="SqlDataSource1"
         ID="GridView1" runat="server">
        </asp:GridView>
        <asp:SqlDataSource
ConnectionString="<%$ ConnectionStrings:Gridview %>"
         SelectCommand="select * from tblHello"
         ID="SqlDataSource1" runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Kết xuất của chương trình




Sử dụng Fields với điều khiển GridView

- BoundField: cho phép bạn hiển thị giá trị của các mục dữ liệu dạng Text
- CheckBoxField:  cho phép bạn hiển thị giá trị của dữ liệu dưới dạng CheckBox.
- CammandField: hiển thị 1 liên kết cho phép chỉnh sửa, xoá hay chọn dòng dữ liệu
- ButtonField: Cho phép hiển thị dữ liệu như một Button(Button, ImageButton, linkButton, Push Button)
- HyperLinkButton: Cho phép hiển thị dữ liệu như một liên kết đến một trang web khác.
- ImagesField: Cho phép bạn hiển thị dữ liệu như một Ảnh
- TemplateField: cho phép bạn hiển thị dữ liệu một cách tuỳ biến với các thẻ HTML hoặc ASP.NET.

# Chương 11 Sử dụng DetailView và FormView

Hai điều khiển này cho phép bạn làm việc với một trường dữ liệu đơn tại mỗi thời điểm

Cả hai điều khiển này cho phép bản thay đổi, thêm mới hay xoá dữ liệu như một bản ghi cơ sở dữ liệu, và nó cho phép bạn chuyển sang trang tiếp theo hay quay lại trang trước thông qua thiết lập dữ liệu.

## I. DetailView

### 1. Hiển thị dữ liệu với DetailView

DetailView được đưa ra hiển thị như một bảng(<Table>) trong HTML để hiển thị dữ liệu một bản ghi.

Ví dụ: Trang DetailView.aspx

Code 11.1

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Detail View</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:DetailsView ID="DetailsView1"
         DataSourceID="SqlDataSource1" runat="server" Height="50px"
Width="125px">
        </asp:DetailsView>
        <asp:SqlDataSource ID="SqlDataSource1"
         ConnectionString="<%$ConnectionStrings:hcubiuData %>"
         SelectCommand="select * from tblIntrodure"
runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Vẫn với cơ sở dữ liệu từ chương trước bạn đưa dữ liệu của bảng tblIntrodure vào SqlDataSource và điền nó vào DetailView1 với thuộc tính DataSourceID của nó

Kết xuất của chương trình sẽ như sau:

| | |
|---|---|
| pkIntrodureID | 1 |
| sTitle | Tiêu đề diễn đàn ĐH Hoa Sen |
| sSummary | Tiêu đề diễn đàn ĐH Hoa Sen |
| sContent | ĐH Hoa Sen |
| iPosition | 1 |

Bạn cũng có thể đưa dữ liệu vào DetailView từ một mảng hay danh sách dữ liệu

Ví dụ:

Bạn tạo một lớp Employee.cs

Code 11.2

```
using System;

public class Employee
{
    private int _PersonID;
    public int PersonID
    {
        get { return _PersonID; }
        set { _PersonID = value; }
    }

    private string _Hoten;
    public string Hoten
    {
        get { return _Hoten; }
        set { _Hoten = value; }
    }

    private int _Tuoi;
    public int Tuoi
    {
        get { return _Tuoi; }
        set { _Tuoi = value; }
    }
      public Employee()
      {
      }

    public Employee(int _PersonID, string _Hoten, int _Tuoi)
    {
        this._PersonID = _PersonID;
        this._Hoten = _Hoten;
        this._Tuoi = _Tuoi;
```

```
    }
}
```

Code 11.3 DetailViewPerson.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Detail View</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:DetailsView ID="DetailsView1" runat="server" Height="50px"
Width="125px">
        </asp:DetailsView>
    </div>
    </form>
</body>
</html>
```

Code 11.4 DetailViewPerson.aspx.cs

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

public partial class DetailViewPerson : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            Employee newEmploy=new Employee(1,"HCUBIU",25);
            List<Employee> listEmploy=new List<Employee>();
            listEmploy.Add(newEmploy);
            DetailsView1.DataSource = listEmploy;
            DetailsView1.DataBind();
        }
    }
}
```

Trong ví dụ này chúng ta tạo ra một lớp Employee và chúng ta đưa dữ liệu vào DetailView1 với thuộc tính DataSource và phương thức DataBind điền dữ liệu vào.

**2. Sử dụng  Fields với điều khiển DetailView**

DetailView hỗ trợ tất cả các Field như GridView

- BoundField: cho phép bạn hiển thị giá trị của dữ liệu như Text
- CheckBoxField: hiển thị dữ liệu dưới dạng một CheckBox

- CommandField: hiển thị liên kết cho phép chỉnh sửa, thêm mới, xoá dữ liệu của dòng.
- ButtonField: hiển thị dữ liệu như một button(ImageButton, )
- HyperLinkField: hiển thị môt liên kết
- ImageField: hiển thị ảnh
- TemplateFile: cho phép hiển thị các điều khiển tuỳ biến.

Ví dụ:

Code 11.5

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Fields</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:DetailsView ID="DetailsView1" AutoGenerateRows="false"
        DataSourceID="SqlDataSource1" runat="server" Height="50px"
Width="125px">
         <Fields>
            <asp:BoundField DataField="pkIntrodureID" HeaderText="ID"
/>
            <asp:BoundField DataField="sTitle" HeaderText="Tiêu đề" />
            <asp:BoundField DataField="iPosition" HeaderText="Vị trí"
/>
         </Fields>
        </asp:DetailsView>
        <asp:SqlDataSource ID="SqlDataSource1"
         ConnectionString="<%$ConnectionStrings:hcubiuData %>"
         SelectCommand="select * from tblIntrodure"
runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Trong ví dụ trên bạn đưa vào 3 BoundField và điền vào dữ liệu với thuộc tính DataField và thiết đặt cho nó tiêu đề với HeaderText, để đưa ra dữ liệu như thế này bạn cần thiết lập thuộc tính AutoGenerateRows=”false”.

Kết xuất của chương trình

| | |
|---|---|
| id | 1 |
| Tiêu đề | Tiêu đề Sinh Viên Hoa Sen |
| Vị trí | 1 |

### 3. Hiển thị DetailView với dữ liệu rỗng

Ví dụ

Code 11.6

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Null Data</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:DetailsView ID="DetailsView1"
         DataSourceID="SqlDataSource1" EmptyDataText="Dữ liệu không có"
runat="server" Height="50px" Width="125px">
        </asp:DetailsView>
        <asp:SqlDataSource ID="SqlDataSource1"
         ConnectionString="<%$ConnectionStrings:hcubiuData %>"
         SelectCommand="select * from tblProduct"
runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Kết xuất của chương trình

| |
|---|
| Dữ liệu không có |

Trong ví dụ trên ta đưa dữ liệu vào DetailView1 với dữ liệu từ bảng tblProduct(chưa được nạp dữ liệu), trong DetailView1 ta thêm vào thuộc tính EmptyDataText="Dữ liệu không có" để khi trong bảng không có dữ liệu chuỗi Text nằm trong thuộc tính EmptyDataText sẽ được đưa ra.

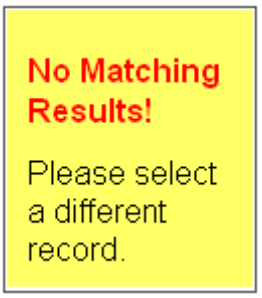
Bạn cũng có thể Customize chuỗi text hiển thị ra khi chưa có nội dung bằng EmptyDataTemple như ví dụ sau:

Ví dụ: DetailViewDatanull.aspx

Code 11.7

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Null Data</title>
    <style type="text/css">
        .noMatch{background-color:#ffff66;padding:10px;font-
family:Arial,Sans-Serif;}
        .noMatch h1{color:red;font-size:16px;font-weight:bold;}
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:DetailsView ID="DetailsView1"
         DataSourceID="SqlDataSource1" runat="server" Height="50px"
Width="125px">
         <EmptyDataTemplate>
            <div class="noMatch">
                <h1>No Matching Results!</h1>
                Please select a different record.
            </div>
         </EmptyDataTemplate>
        </asp:DetailsView>
        <asp:SqlDataSource ID="SqlDataSource1"
         ConnectionString="<%$ConnectionStrings:hcubiuData %>"
         SelectCommand="select * from tblProduct"
runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Kết xuất của chương trình sẽ như sau:



# Chuong 12. Sử dụng Repeater và DataList

Cả hai điều khiển này đều cho phép hiển thị tập hợp các mục dữ liệu tại một thời điểm, nói cách khác là có thể hiển thị tất cả các dòng trong bảng dữ liệu.

## I. sử dụng điều khiển Repeater

### 1. Hiển thị dữ liệu với Repeater

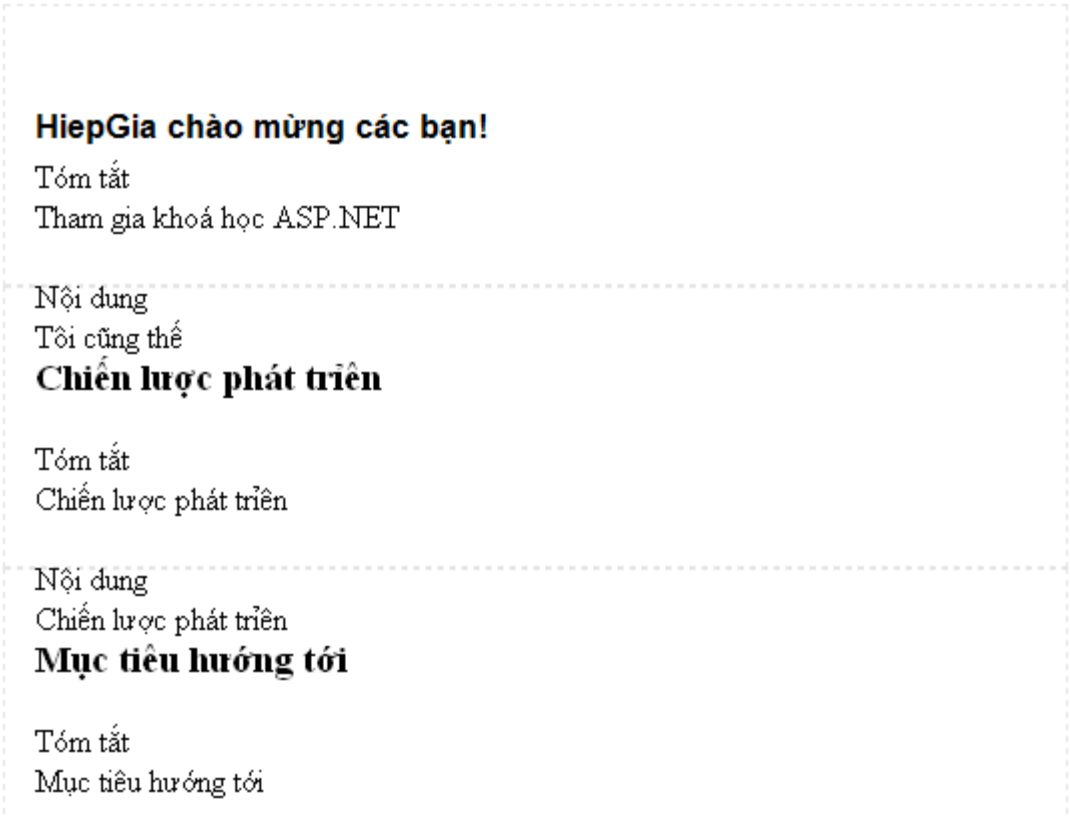
Để hiển thị dữ liệu với Repeater bạn phải tạo một ItemTemplate

Ví dụ: trang Repeater.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Repeater</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:Repeater DataSourceID="SqlDataSource1" ID="Repeater1"
runat="server">
            <ItemTemplate>
                <div class="navpage">
                    <h3><%#Eval("sTitle") %></h3>
                    Tóm tắt<br /> <%#Eval("sSummary")%><br /><br />
                    Nội dung<br />
                    <%#Eval("sContent") %>
                </div>
            </ItemTemplate>
        </asp:Repeater>
        <asp:SqlDataSource ID="SqlDataSource1"
ConnectionString="<%$ConnectionStrings:hcubiuData %>"
        SelectCommand="select sTitle,sSummary,sContent from
tblIntrodure" runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Bạn đưa điều khiển Repeater vào Form và đặt thuộc tính DataSourceID=”SqlDataSource1” và thêm vào một ItemTemplate trong này bạn có thể điều chỉnh cách hiển thị dữ liệu theo ý muốn của bạn với các thẻ của Asp.Net hoặc HTML.

Kết xuất của đoạn Code trên sẽ như sau:




**2. Sử dụng Template với điều khiển Repeater**

Điều khiển Repeater hỗ trợ 5 kiểu của Templates

- ItemTemplate: định dạng mỗi item từ nguồn dữ liệu
- AlternatingItemTemplate: định dạng tất cả các item dữ liệu khác từ nguồn dữ liệu
- SeparatorTemplate: định dạng giữa hai item từ nguồn cơ sở dữ liệu
- HeaderTemplate: Định dạng header cho tất cả các item
- FooterTemplate: Định dạn Footer cho tất cả các item.

Ví dụ: trang RepeaterDP.aspx

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

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

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

```
    <style type="text/css">
        html{background-color:silver;}
        .content{width:600px;border:solid 1px black;background-
color:white;}
        .intro{border-collapse:collapse;}
        .intro th,.intro td{padding:10px;border-bottom:1px solid
black;}
        .alternating{background-color:#eeeeee;}
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:Repeater DataSourceID="SqlDataSource1" ID="Repeater1"
runat="server">
            <HeaderTemplate>
                <table class="intro">
                    <tr>
                        <th>Tiêu đề</th>
                        <td>Tóm tắt</td>
                    </tr>
            </HeaderTemplate>
            <ItemTemplate>
                <tr>
                    <td><%#Eval("sTitle") %></td>
                    <td><%#Eval("sSummary")%></td>
                </tr>
            </ItemTemplate>
            <AlternatingItemTemplate>
                <tr class="alternating">
                    <td><%#Eval("sTitle") %></td>
                    <td><%#Eval("sSummary") %></td>
                </tr>
            </AlternatingItemTemplate>
            <FooterTemplate>
                </table>
            </FooterTemplate>
        </asp:Repeater>
        <asp:SqlDataSource ID="SqlDataSource1"
ConnectionString="<%$ConnectionStrings:hcubiuData %>"
        SelectCommand="select sTitle,sSummary,sContent from
tblIntrodure" runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Kết xuất của chương trình:

| Tiêu đề | Tóm tắt |
|---|---|
| HiepGia chào mừng các bạn | Tham gia khoá học ASP.NET |
| Chiến lược phát triển | Chiến lược phát triển |
| Mục tiêu hướng tới | Mục tiêu hướng tới |
| Lĩnh vực hoạt động | Lĩnh vực hoạt động |

### 3. Điều khiển Repeater với các sự kiện

Điểu khiển Repeater hỗ trợ các sự kiện sau:

- DataBinding: xảy ra khi Repeater được rạng buộc đến dữ liệu
- ItemCommand: xảy ra khi bên trong Repeater chứa đựng điều khiện Command và điều khiển này đưa ra sự kiện.
- ItemCreate: xảy ra khi mỗi RepeaterItem được tạo
- ItemDataBound: xảy ra khi mỗi item của Repeater được ràng buộc

Ví dụ trang RepeaterEvent.aspx

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

<script runat="server">
    void Repeater1_ItemDataBound(object sender, RepeaterItemEventArgs
e)
    {
        Response.Write("<li>Dữ liệu đang được tạo</li>");
    }

    void repeater1_DataBinding(object sender, EventArgs e)
    {
        Response.Write("Ràng buộc dữ liệu cho Repeater");
    }

    void Repeater1_ItemCommand(object sender, RepeaterCommandEventArgs
e)
    {
        switch (e.CommandName)
        {
            case "insert":
                Response.Write(e.CommandArgument.ToString() + ": Bạn
chọn xoá dữ liệu");
                break;
            case "update":
                Response.Write(e.CommandArgument.ToString() + ": Bạn
chọn cập nhật dữ liệu");
                break;
            case "delete":
```

```
                Response.Write(e.CommandArgument.ToString() + ": Bạn
chọn xoá dữ liệu");
                break;
        }
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Repeater</title>
    <style type="text/css">
        html{background-color:silver;}
        .content{width:600px;border:solid 1px black;background-
color:white;}
        .intro{border-collapse:collapse;}
        .intro th,.intro td{padding:10px;border-bottom:1px solid
black;}
        .alternating{background-color:#eeeeee;}
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:Repeater OnItemDataBound="Repeater1_ItemDataBound"
OnDataBinding="repeater1_DataBinding"
OnItemCommand="Repeater1_ItemCommand" DataSourceID="SqlDataSource1"
ID="Repeater1" runat="server">
            <HeaderTemplate>
                <table class="intro">
                    <tr>
                        <th>Tiêu đề</th>
                        <td>Tóm tắt</td>
                    </tr>
            </HeaderTemplate>
            <ItemTemplate>
                <tr>
                    <td><%#Eval("sTitle") %></td>
                    <td><%#Eval("sSummary")%></td>
                </tr>
                <tr>
                    <td colspan="2">
                        <asp:LinkButton ID="lbninsert"
CommandName="insert" CommandArgument="insert"
runat="server">Insert</asp:LinkButton> |
                        <asp:LinkButton ID="lbnupdate"
CommandName="update" CommandArgument="update"
runat="server">Update</asp:LinkButton> |
                        <asp:LinkButton ID="lbndelete"
CommandName="delete" CommandArgument="delete"
runat="server">Delete</asp:LinkButton>
                    </td>
                </tr>
            </ItemTemplate>
            <AlternatingItemTemplate>
                <tr class="alternating">
```

```
                    <td><%#Eval("sTitle") %></td>
                    <td><%#Eval("sSummary") %></td>
                </tr>
                <tr>
                    <td colspan="2">
                        <asp:LinkButton ID="lbninsert"
CommandName="insert" CommandArgument="insert"
runat="server">Insert</asp:LinkButton> |
                        <asp:LinkButton ID="lbnupdate"
CommandName="update" CommandArgument="update"
runat="server">Update</asp:LinkButton> |
                        <asp:LinkButton ID="lbndelete"
CommandName="delete" CommandArgument="delete"
runat="server">Delete</asp:LinkButton>
                    </td>
                </tr>
            </AlternatingItemTemplate>
            <FooterTemplate>
                </table>
            </FooterTemplate>
        </asp:Repeater>
        <asp:SqlDataSource ID="SqlDataSource1"
ConnectionString="<%$ConnectionStrings:hcubiuData %>"
        SelectCommand="select sTitle,sSummary,sContent from
tblIntrodure" runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Ví dụ trên bạn đưa vào 3 sự kiện cho điều khiển Repeater, tương ứng với nó là 3 sự kiện được kích hoạt: sự kiện

Repeater1_ItemDataBound được đưa ra làm và nó được thực hiện mỗi khi dữ liệu hay một item được đưa vào Repeater với ví dụ trên nó sẽ in ra "dữ liệu đang được tạo" x(số hàng trong bảng dữ liệu) lần.

repeater1_DataBinding thực hiện công việc khi dữ liệu được điền vào Repeater.

Repeater1_ItemCommand: phụ thuộc vào tên CommandName để đưa ra công việc thích hợp(nếu chưa rõ CommandName bạn xem lại phần Các điều khiển cơ bản của ASP.NET)

## II. Sử dụng điều khiển DataList

### 1. Hiển thị dữ liệu với DataList

Ví dụ: trang DataList.aspx

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

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

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

```
    <title>Repeater</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:DataList ID="DataList1" DataSourceID="SqlDataSource1"
runat="server">
            <ItemTemplate>
                <h3><%#Eval("sTitle") %></h3>
                Tóm tắt: <%#Eval("sSummary") %>
            </ItemTemplate>
        </asp:DataList>
        <asp:SqlDataSource ID="SqlDataSource1"
ConnectionString="<%$ConnectionStrings:hcubiuData %>"
        SelectCommand="select sTitle,sSummary,sContent from
tblIntrodure" runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

## 2. Hiển thị dữ liệu trong nhiều cột

Bạn có thể hiển thị dữ liệu trong điều khiển DataList trong nhiều cột giống như mỗi item nằm trong một Cells với 2 thuộc tính của Repeater là:

- RepeaterColumns: số cột hiển thị
- RepeateDirection: hướng để hiển thị các ô. Có thể giá trị là Horizontal hoặc Verical

Ví dụ sau đây sẽ hiển thị dữ liệu trong DataList với 3 cột

Ví dụ trang DataListMutilColumn.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Repeater</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:DataList ID="DataList1" GridLines="Both" RepeatColumns="3"
DataSourceID="SqlDataSource1" runat="server">
            <ItemTemplate>
                <h3><%#Eval("sTitle") %></h3>
                Tóm tắt: <%#Eval("sSummary") %>
            </ItemTemplate>
        </asp:DataList>
        <asp:SqlDataSource ID="SqlDataSource1"
ConnectionString="<%$ConnectionStrings:hcubiuData %>"
        SelectCommand="select sTitle,sSummary,sContent from
tblIntrodure" runat="server"></asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Ở đây bạn chỉ cần thêm vào hai thuộc tính RepeaterColumns=”3” và GridLines=”Both”

kết xuất của chương trình sau:

| | | |
|---|---|---|
| **HiepGia chào các bạn**<br>Tóm tắt: Tham gia khoá học ASP.NET | **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động | **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động |
| **Chiến lược phát triển**<br>Tóm tắt: Chiến lược phát triển | **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động | **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động |
| **Mục tiêu hướng tới**<br>Tóm tắt: Mục tiêu hướng tới | **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động | **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động |
| **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động | **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động | **Lĩnh vực hoạt động**<br>Tóm tắt: Lĩnh vực hoạt động |

### 3. Sử dụng Template với điều khiển DataList

DataList hỗ trợ tất cả các templates giống với Repeater và nó được thêm vào hai templates:

- EditItemTemplate: hiển thị khi dòng được chọn để chỉnh sửa
- SelectedItemTemplate: được hiển thị khi 1 dòng được lựa chọn

### 4. Chọn dữ liệu với điều khiển DataList

Bạn có thể sử dụng DataList như một menu bằng việc thêm vào thuộc tính SelectedValue

Ví dụ: trang DataListselect.aspx

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

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Repeater</title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="navcontain">
        <asp:DataList ID="DataList1" DataKeyField="pkIntrodureID"
DataSourceID="SqlDataSource1" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lbnselect" CommandName="Select"
runat="server" Text='<%#Eval("sTitle") %>' />
```

```
            </ItemTemplate>
        </asp:DataList>
        <asp:DataList ID="datalist2" runat="server"
DataSourceID="SqlDataSource2">
            <ItemTemplate>
                <h3><%#Eval("sTitle") %></h3>
                Tóm tắt: <%#Eval("sSummary") %><br /><br />
                Nội dung: <%#Eval("sContent") %>
            </ItemTemplate>
        </asp:DataList>

        <asp:SqlDataSource ID="SqlDataSource1"
ConnectionString="<%$ConnectionStrings:hcubiuData %>"
        SelectCommand="select pkIntrodureID,sTitle,sSummary,sContent
from tblIntrodure" runat="server"></asp:SqlDataSource>
        <asp:SqlDataSource ID="SqlDataSource2"
ConnectionString="<%$ConnectionStrings:hcubiuData %>"
        SelectCommand="select pkIntrodureID,sTitle,sSummary,sContent
from tblIntrodure where pkIntrodureID=@pkIntrodureID" runat="server">
            <SelectParameters>
                <asp:ControlParameter Name="pkIntrodureID"
ControlID="DataList1" PropertyName="SelectedValue" />
            </SelectParameters>
        </asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Trong ví dụ trên Datalist1 dùng làm menu trong ItemTemplate chúng ta đưa vào một LinkButton và cung cấp cho nó thuộc tính CommandName=”Select”. Ta tiếp tục đưa thêm vào một SqlDataSource2 với việc đưa vào một tham số nhận giá trị về từ Datalist1 với tham số truyền pkIntrodureID, và chúng ta đưa thêm vào một Datalist2 với DataSourceID=”SqlDataSource2”. Như vậy khi chạy chương trình bạn nhấn vào mỗi mục trong Datalist1 thì dữ liệu đầy đủ tương ứng sẽ hiện trong trong DataList2

Thay đổi dữ liệu với điều khiển DataList

Bạn có thể sử dụng DataList để thay đổi dữ liệu bản ghi. Tuy nhiên để chỉnh sửa dữ liệu của bản ghi nó yêu cầu nhiều phải viết nhiều code hơn so với các điều khiển ràng buộc dữ liệu khác như GridView, FormView hay DetailView.

Ví dụ sau đây sẽ hương dẫn bạn cách thay đổi và xoá dữ liệu từ DataList.

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

<script runat="server">
    void DataList1_EditCommand(object sender, DataListCommandEventArgs
e)
    {
        DataList1.EditItemIndex = e.Item.ItemIndex;
        DataList1.DataBind();
    }

    void DataList1_DeleteCommand(object sender,
DataListCommandEventArgs e)
    {
```

```
        SqlDataSource1.DeleteParameters["pkIntrodureID"].DefaultValue =
DataList1.DataKeys[e.Item.ItemIndex].ToString();
        SqlDataSource1.Delete();
    }

    void DataList1_UpdateCommand(object sender,
DataListCommandEventArgs e)
    {
        TextBox txtTitle = (TextBox)e.Item.FindControl("txtTitle");
        TextBox txtSummary = (TextBox)e.Item.FindControl("txtSummary");
        SqlDataSource1.UpdateParameters["pkIntrodureID"].DefaultValue =
DataList1.DataKeys[e.Item.ItemIndex].ToString();
        SqlDataSource1.UpdateParameters["sTitle"].DefaultValue =
txtTitle.Text;
        SqlDataSource1.UpdateParameters["sSummary"].DefaultValue =
txtSummary.Text;
        SqlDataSource1.Update();
        DataList1.EditItemIndex = -1;
    }
    void DataList1_CancelCommand(object sender,
DataListCommandEventArgs e)
    {
        DataList1.EditItemIndex = -1;
        DataList1.DataBind();
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Repeater</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:DataList ID="DataList1" DataSourceID="SqlDataSource1"
        DataKeyField="pkIntrodureID" GridLines="None"
        OnEditCommand="DataList1_EditCommand"
        OnDeleteCommand="DataList1_DeleteCommand"
        OnUpdateCommand="DataList1_UpdateCommand"
        OnCancelCommand="DataList1_CancelCommand"
        runat="server">
        <ItemTemplate>
         <h3><%#Eval("sTitle") %></h3>
         <%#Eval("sSummary") %><br /><br />
         <asp:LinkButton ID="lbnedit" runat="server" Text="Edit"
CommandName="Edit" /> |
         <asp:LinkButton ID="lbndelete" runat="server" Text="Delete"
OnClientClick="return confirm('Bạn có chắc chắn xoá mục này không?');"
CommandName="Delete" />
        </ItemTemplate>
        <EditItemTemplate>

            Tiêu đề:<br />
            <asp:TextBox ID="txtTitle" runat="server"
Text='<%#Eval("sTitle") %>'/><br /><br />
```

```
            Tóm tắt:<br />
            <asp:TextBox ID="txtSummary" runat="server"
Text='<%#Eval("sSummary") %>' /><br /><br />
            <asp:LinkButton ID="lbnUpdate" runat="server"
CommandName="Update" Text="Update" /> |
            <asp:LinkButton ID="lbncancel" runat="server"
CommandName="Cancel" Text="Cancel" />

        </EditItemTemplate>
        </asp:DataList>
        <asp:SqlDataSource ID="SqlDataSource1"
ConnectionString="<%$ConnectionStrings:hcubiuData %>"
        UpdateCommand="Update tblIntrodure set sTitle=@sTitle,
sSummary=@sSummary where pkIntrodureID=@pkIntrodureID"
        DeleteCommand="Delete from tblIntrodure where
pkIntrodureID=@pkIntrodureID"
        SelectCommand="select * from tblIntrodure" runat="server">
            <UpdateParameters>
                <asp:Parameter Name="pkIntrodureID" />
                <asp:Parameter Name="sTitle" />
                <asp:Parameter Name="sSummary" />
            </UpdateParameters>
            <DeleteParameters>
                <asp:Parameter Name="pkIntrodureID" />
            </DeleteParameters>
        </asp:SqlDataSource>
    </div>
    </form>
</body>
</html>
```

Cách thao tác dữ liệu với DataList gần giống với FormView.

Để làm được việc này bạn cần cung cấp các button có CommandName  ứng với các phương thức  mà được DataList hỗ trợ . cùng với việc đưa vào các tham số tương ứng trong thành phần UpdateParameter và DeleteParameter của SqlDataSource.

# Chương 13: Trạng thái

## I. Sử dụng Cookie

### Cookie làm việc như thế nào?

Khi trình duyệt web tạo một Cookie, một nội dung sẽ được lưu vào header của trang web với nội dung giống như sau:

Set-Cookie: Message=Hello

Phần tiêu đề Set-Cookie này gây ra cho trình duyệt web tạo một Cookie có tên là Message và giá trị của nó là Hello.

Sau khi một Cookie được tạo trên trình duyệt, Mỗi khi trình duyệt yêu cầu một trang web trong ứng dụng, trình duyệt sẽ gửi một header có dạng giống như sau:

Cookie: Message=Hello

Tiêu đề Cookie chứa đựng tất cả các Cookie mà được tạo trên Web Server. Cookie được gửi trở lại mỗi khi một yêu cầu được đưa ra trên trình duyệt web.

Bạn có thể tạo hai kiểu của Cookie, Session Cookies và Persistent Cookies. Session cookies chỉ tồn tại trong bộ nhớ khi trình duyệt web bị đóng lại thì nó cũng bị xóa đi.

Còn Persistent Cookies có thể tồn tại hàng tháng hoặc hàng năm. Khi bạn tạo một Persistent Cookies, nó sẽ được lưu trữ trên web browse trên máy tính của bạn. với IE ví dụ nó sẽ được lưu trữ trong một file Text theo thư mục

\Documents and Settings\[user]\Cookies

Còn với FireFox nó lưu trữ trong thư mục theo đường dẫn sau:

\Documents and Settings\[user]\Application Data\Mozilla\Firefox\Profiles\[random

folder name]\Cookies.txt

bởi vì sự lưu trữ cookies trên các trình duyệt khác nhau để ở các thư mục khác nhau lên khi bạn tạo Cookies trên IE thì nó sẽ không tồn tại trên FireFox và ngược lại.

### Bảo mật với Cookie

### Tạo Cookies

Bạn có thể tạo cookies với câu lệnh Response.Cookies, tất cả các Cookies sẽ được gửi từ Web Server đến Web Browser.

Ví dụ sau đây sẽ tạo ra một Cookies Message với giá trị được lấy từ hộp TextBox trên Form

Ví dụ 1: Trang setCookies.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default.aspx.cs" Inherits="_Default" %>
<script runat="server">
    protected void Add_Click(object sender, EventArgs e)
    {
        Response.Cookies["Message"].Value = txtCookies.Text;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

```
<head runat="server">
    <title>Create Cookies</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="Label1" runat="server" Text="Cookie
Value"></asp:Label>
        <asp:TextBox ID="txtCookies" runat="server"></asp:TextBox>
        <asp:Button ID="Add" runat="server" OnClick"Add_Click"
Text="Button" />
    </div>
    </form>
</body>
</html>
```

Trong ví dụ một là chúng ta tạo ra một Session Cookies, còn nếu bạn muốn tạo một Persistent Cookies bạn cần chỉ định thời hạn kết thúc cho Cookies .

Ví dụ 2 trang setPersistentCookies.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="setPersistentCookies.aspx.cs"
Inherits="setPersistentCookies" %>
<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        int counter=0;
        if (Request.Cookies["counter"] != null)
            counter = Int32.Parse(Request.Cookies["counter"].Value);
        counter++;
        Response.Cookies["counter"].Value = counter.ToString();
        Response.Cookies["counter"].Expires = DateTime.Now.AddYears(2);
        this.Label1.Text = Response.Cookies["counter"].Value;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Set Persitent Cookies</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Trong ví dụ trên khi chạy chương trình mỗi lần bạn Refresh lại trang thì giá trị của Label1 sẽ tăng lên một. Và với câu lệnh Response.Cookies[“counter”].Expires=Datetime.Now.AddYears(2), có nghĩa là thời gian tồn tại của Cookie này sẽ là 2 năm.

**Đọc dữ liệu từ Cookies**

Bạn sử dụng lện Request.Cookies để lấy dữ liệu từ Cookies, bạn xem lại ví dụ 2 trang setPersistentCookies.aspx.

Khi bạn có một tập hợp các Cookies bạn có thể lấy tất cả giá trị của các Cookies trên website của mình, ví dụ sau đây sẽ hướng dẫn bạn làm việc đó.

Ví dụ 3 trang GetallCookies

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="GetAllCookies.aspx.cs" Inherits="GetAllCookies" %>
<script runat="server">
    void Page_Load()
    {
        ArrayList colCookies = new ArrayList();
        for (int i = 0; i < Request.Cookies.Count; i++)
            colCookies.Add(Request.Cookies[i]);
        grdCookies.DataSource = colCookies;
        grdCookies.DataBind();
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Get All Cookies</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:GridView ID="grdCookies" runat="server">
        </asp:GridView>
    </div>
    </form>
</body>
</html>
```

**Thiết lập thuộc tính cho Cookies**

Cookies được đưa ra với lớp HttpCookie, khi bạn tạo hoặc lấy giá trị từ một Cookie có thể bạn sẽ sử dụng một vài thuộc tính của lớp này:

**- Domain**: cho phép bạn chỉ định Domain kết hợp với Cookie. Giá trị mặc định là Domain hiện tại.
- **Expires**: Cho phép tạo Persistent Cookies với thời hạn được chỉ định
- **HasKeys**: Cho phép bạn định rõ Cookies có nhiều giá trị hay không.
- **HttpOnly**: Cho phép đưa ra một Cookies từ JavaScript.
- **Name**: chỉ định tên cho Cookies.
+- **Path**: Cho phép chỉ định đường dẫn kết hợp với Cookies. Giá trị mặc định là /.
- **Secure:** Cho phép một Cookie được chuyển tác ngang tới kết nối Sercure Sockets Layer (SSL).
- **Value:** Cho phép lấy hoặc thiết lập giá trị cho Cooki
**- Values**: Cho phép bạn lấy hoặc thiết lập giá trị riêng khi làm việc với Cookies có nhiều giá trị.

**Xóa Cookies**

Để xóa một Cookie bạn thiết lập ngày hết hạn cho Cookies là -1

Ví dụ như câu lệnh dưới đây:

Response.Cookies[“Message”].Expires = DateTime.Now.AddDays(-1);

Trên ví dụ trên chúng ta sẽ xóa Cookie vơi tên là Message.

**Làm việc với Cookies nhiều giá trị:**

Trong trình duyệt không lên lưu trưc hơn 20 Cookies từ một Domain, thay vào đó bạn có thể làm việc với một Cookie nhiều giá trị.

Một Cookies nhiều giá trị là một Cookies đơn chứa đựng nhiều khóa con, bạn có thể tạo nhiều khóa con như bạn muốn.

Như ví dụ dưới đây bạn tạo ra một Cookies Person nhiều giá trị, Cookie Person lưu trữ các giá trị Họ tên, Ngày sinh và màu sắc yêu thích.

Ví dụ 4 trang SetCookieValues.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="SetCookieValues.aspx.cs" Inherits="SetCookieValues" %>
<script runat="server">
    protected void btnsubmit_Click(object sender, EventArgs e)
    {
        Response.Cookies["Person"]["Hoten"] = txtHoten.Text;
        Response.Cookies["Person"]["Ngaysinh"] = txtNgaysinh.Text;
        Response.Cookies["Person"]["Color"] = txtColor.Text;
        Response.Cookies["Person"].Expires = DateTime.MaxValue;
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Set Cookie MutilValues</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <table>
            <tr>
                <td>Họ tên</td>
                <td><asp:TextBox ID="txtHoten" runat="server" /></td>
            </tr>
            <tr>
                <td>Ngày sinh</td>
                <td><asp:TextBox ID="txtNgaysinh" runat="server"
/></td>
            </tr>
            <tr>
                <td>Màu yêu thích</td>
                <td><asp:TextBox ID="txtColor" runat="server" /></td>
            </tr>
            <tr>
```

```
                <td colspan="2">
                    <asp:Button ID="btnsubmit" runat="server"
Text="Submit" OnClick="btnsubmit_Click" />
                </td>
            </tr>
        </table>
    </div>
    </form>
</body>
</html>
```

Việc lấy giá trị của Cookie nhiều giá trị tương tự như các phần trên, học viên về nhà hoàn thiện nốt.

## II. Làm việc với Session

Bạn có thể chưa thực sự dùng Cookies để lưu trữ Shoping Cart. Một Cookie vừa quá nhỏ và quá đơn giản. Để làm việc ngoài giới hạn của Cookie ASP.NET Framework hỗ trợ một chức năng mới được gọi là Session State

Giống với Cookie Session lưu trữ dữ liệu trong phạm vi riêng với từng người sử dụng. Nhưng không giống với Cookie Session không giới hạn dung lượng, nếu bạn cần bạn có thể lưu trữ hàng Gigabyte dữ liệu, hơn thế nữa Session có thể đưa ra điều đối tượng phức tạp hơn là chuỗi Text. Bạn có thể lưu trữ một vài đối tượng trong Session. Ví dụ bạn có thể lưu trữ một Dataset hay một Shoping cart trong Session.

1. thêm dữ liệu vào Session

Bạn thêm dữ liệu vào trạng thái Session bằng việc sử dụng đối tượng Session, Ví dụ sau đây sẽ thêm một dữ liệu vào Session có tên là Message và giá trị của nó là "Hello World"

Ví dụ 1: Trang Sessionset.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Sessionset.aspx.cs" Inherits="_Default" %>
<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        Session["Message"] = "Hello World";
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Session set</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <h1>Session state item added!</h1>
    </div>
    </form>
</body>
</html>
```

2. Lấy dữ liệu từ một Session

Ví dụ: Trang Sessionset.aspx

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

<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        lblsession.Text = Session["Message"].ToString();
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Session get</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="lblsession" runat="server" />
    </div>
    </form>
</body>
</html>
```

Bạn lưu ý rằng cũng như Cookie khi một Session được tạo ra, một trạng thái session có tên ASP.NET_SessionID được tự động thêm vào trình duyệt của bạn và Session này được lưu trữ trên web server và không lưu trữ trên webClient. Và khi bạn tắt trình duyệt đi thì Session này của bạn vẫn tồn tại trong khoảng thời gian quy định, mà ASP.NET Framework quy định thời gian mặc định của Session là 20 phút. bạn có thể thiết lập thời gian nhiều hơn.

3. Lưu trữ cơ sở dữ liệu trong Session

Bạn có thể tạo Session state để tạo một vùng nhớ cho người sử dụng. ví dụ bạn có thể tải dữ liệu cho một người sử dụng và cho phép người sử dụng đó sắp xếp hay lọc dữ liệu.

4. Sử dụng đối tượng Session

Chương trình ứng dụng chính giao tiếp và làm việc với Session State là lớp HttpSessionState. Đối tượng này được thể hiện bởi các thuộc tính Page.Session, Context.Session, UserControl.Session, Webservice.Session và Application.Session. có nghĩa là bạn có thể làm việc với Session bất kỳ đâu trong ứng dụng web.

Lớp HttpSessionState hỗ trợ các thuộc tính sau:

- CookieMode: có cho phép Cookie Session hay không?
- Count: cho phép lấy số dữ liệu trong Session State
- IsCookieless: Cho phép chỉ rõ có cho phép Cookieless hay không?
- **IsNewSession**—Enables you to determine whether a new user session was created

- with the current request.
- **IsReadOnly**—Enables you to determine whether the Session state is read-only.
- **Keys**—Enables you to retrieve a list of item names stored in Session state.
- **Mode**—Enables you to determine the current Session state store provider. Possible
- values are Custom, InProc, Off, SqlServer, and StateServer.
- **SessionID**—Enables you to retrieve the unique session identifier.
- **Timeout**—Enables you to specify the amount of time in minutes before the web
- server assumes that the user has left and discards the session. The maximum value is 525,600 (1 year).

Đối tượng HttpSessionState hỗ trợ các phương thức sau:

- Abandon: Cho phép kết thúc Session của một người sử dụng.
- Clear: Cho phép xoá toàn bộ dữ liệu trong Session State.
- Remove: cho phép bạn xoá từng phần tử trong Session State

5. Điều khiển sự kiện Session

Có hai sự kiện có liên quan với Session State mà bạn có thể điều khiển nó trong file Global.asax là sự kiện Session_Start và Session_End.

Session_Start xảy ra khi một Sesion mới của người sử dụng được tạo ra. Bạn có thể sử dụng sự kiện này để load thông tin của người sử dụng ra từ cơ sở dữ liệu. Ví dụ bạn có thể tải dữ liệu về Shoping cart của người sử dụng trong sự kiện này .

Session_End xảy ra khi kết thúc Session, một Session kết thúc khi thời hạn của Session hết hoặc bởi viecj chỉ định của phương thức Session.Abadon. Ví dụ Khi bạn muốn tự động ghi giỏ hàng của người sử dụng vào bảng dữ liệu trong cơ sở dữ liệu khi Session_End xảy ra.

6. Điều khiển khi Session quá hạn

Mặc định ASP.NET Framework quy định thời gian quá hạn của Session là 20 phút, trong nhiều trường hợp bạn thấy như vậy là quá ít, và bạn nghĩ rằng bạn cần thay đổi thời gian này.

Ví dụ trong trường hợp bạn tạo một trang quản trị của website, khi cập nhật dữ liệu bạn có một bài viết dài, và thời hạn 20 phút không thể đủ thời gian cho bạn cập nhật tin đó, và để hoàn thành bài đó có thể bạn phải mất 1 giờ.

Sự bất lợi là nếu tăng thời gian quá hạn của Session lên thì bộ nhớ của ứng dụng càng phải sử dụng nhiều, vì vậy khi bạn tăng thời hạn của Session thì bộ nhớ của Server sẽ phải dùng càng nhiều.

Tuy nhiên nếu cần bạn vẫn có thể tăng thời hạn của Session bằng cách bạn có thể chỉ định thời gian quá hạn của Session trong file web.config

Ví dụ:

```
<configuration>
    <system.web>
```

```
        <sessionState timeout="60" />
      </system.web>
    </configuration>
```

7. sử dụng Cookieless Session State

Mặc định Session State dựa trên Cookie. ASPNET Framework sử dụng ASP.NET_SessionId Cookie để định danh người sử dụng trên website mà dữ liệu có thể được kết hợp với người sử dụng, nếu người sử dụng vô hiệu hóa Cookie trên trình duyệt thì Session State sẽ không làm việc.

Để Session có thể làm việc khi trình duyệt vô hiệu hóa Cookie bạn cần thêm vào Cookieless Session. Khi Cookieless Session được cho phép, thì Session ID của người sử dụng sẽ được thêm vào trang URL.

Đây là một ví dụ của trang URL nhìn giống với khi Cookieless Session được cho phép.

http://localhost:4945/Original/(S(5pnh11553sszre45oevthxnn))/SomePage.aspx

Bạn cho phép Cookieless Session bằng việc chỉnh sửa các thành phần SessionState trong file web.config. Thành phần SessionState bao gồm các một đặc tính cookieless mà nó chấp nhận các giá trị sau:

- AutoDetect: SessionID được lưu trữ trong một cookie khi trình duyệt có cho phép Cookie. Ngược lại thì nó lưu trữ vào địa chỉ URL.
- UseCookies: Session ID luôn luôn lưu trữ trong cookie
- UseDeviceProfile: Session ID lưu trữ trong cookie khi trình duyệt hỗ trợ Cookie, trường hợp ngược lại nó lưu trữ trong địa chỉ URL.
- UseUri: Session ID luôn luôn được thêm vào URL.

Trong ví dụ sau đây chúng ta thiết lập cookieless là AutoDetect như vậy ASP.NET Framework sẽ kiểm tra sự tồn tại cảu HTTP Cookie header, nếu Cookie header được tìm thấy thì Framework sẽ lưu trữ Session trong một cookie và ngược lại thì nó sẽ thêm vào URL.

Ví dụ:

```
<configuration>
    <system.web>
      <sessionState cookieless="AutoDetect"
regenerateExpiredSessionId="true"/>
    </system.web>
</configuration>
```

## III. Sử dụng Profiles

ASP.NET Framework cung cấp cho bạn một thay thế cho Session hay cookie để lưu trữ thông tin của người sử dụng đó là đối tượng Profile

Bạn tạo một Profile bằng cách định nghĩa một danh sách các thuộc tính Profile trong ứng dụng ờ file web.config trong thư mục root. ASP.NET Framework tự động biên dịch một lớp chứa đựng các thuộc tính này.

Ví dụ sau đây sẽ đưa ra một Profile với ba thuộc tính: firstName, lastName và NumberOfVisits:

Trang web.config

```
<?xml version="1.0"?>

<configuration>
    <system.web>
      <profile>
        <properties>
          <add name="firstName" />
          <add name="lastName"/>
          <add name="NumberOfVisits" type="Int32" defaultValue="0"/>
        </properties>
      </profile>
    </system.web>
</configuration>
```

Khi làm việc với Profile bạn  chú ý một số thuộc tính sau:

- Name: chỉ định tên của thuộc tính.
- Type: cho phép chỉ định kiểu dữ liệu của thuộc tính
- Defaultvalue: cho phép chỉ định giá trị mặc định của thuộc tính
- ReadOnly: cho phép tạo thuộc tính chỉ đọc
- serializeAs: Enables you to specify how a property is persisted into a static repre-
- sentation. Possible values are Binary, ProviderSpecific, String, and Xml.
- allowAnonnyMous: cho phép người sử dụng nặc danh đọc và thiết lập thuộc tính
- Provider: Cho phép bạn kết hợp thuộc tính với một Profile Provider riêng biệt.
- customeProviderData:Enables you to pass custom data to a Profile provider.

Sau khi định nghĩa một Profile trong web.config, bạn có thể sử dụng đối tượng Provider để chỉnh sửa các thuộc tính Profile. Như ví dụ sau đây bạn sẽ chỉnh sửa hai thuộc tính firstName và lastName trên Form, hơn thế nữa chúng ta sẽ thấy mỗi lần trang web được load lại thì giá trị của NumberOfVisit sẽ tăng lên một.

Ví dụ: trang showProfile.aspx

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

<script runat="server">

    void Page_PreRender()
    {
        firstName.Text = Profile.firstName;
        lastName.Text = Profile.lastName;

        Profile.NumberOfVisits++;
        numbervisit.Text = Profile.NumberOfVisits.ToString();
    }

    protected void btnUpdate_Click(object sender, EventArgs e)
    {
        Profile.firstName = txtFirstName.Text;
        Profile.lastName = txtLastName.Text;
```

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

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>show Profile</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
     FirstName: <asp:Label ID="firstName" runat="server" /><br />
     LastName: <asp:Label ID="lastName" runat="server" /><br />
     Number of Visit: <asp:Label ID="numbervisit" runat="server" /><hr
/>
     <asp:Label ID="lblfistname" runat="server" Text="FirstName:" />
     <asp:TextBox ID="txtFirstName" runat="server" /><br />
     <asp:Label ID="lbllastName" runat="server" Text="LastName:" />
     <asp:TextBox ID="txtLastName" runat="server" /><br />
     <asp:Button ID="btnUpdate" Text="Update" runat="server"
OnClick="btnUpdate_Click" />
    </div>
    </form>
</body>
</html>
```

Và kết xuất của chương trình như sau:


Chú ý rằng thuộc tính của Profile được trình bày với kiểu dữ liệu đã quy định trong thuộc tính type mà ta định nghĩa trong Profile.

Một thuộc tính quan trọng của Profile nó là nó có khả năng giữ lại giá trị của thuộc tính khi người sử dụng rời khỏi trang web, ví dụ nếu bạn gán thuộc tính Profile cho một người sử dụng, người sử dụng đó không quay lại website trong 500 năm thì giá trị đó vẫn được giữ lại cho người sử dụng.

Đối tượng Profile sử dụng models Provider. mặc định Profile Provider là SqlProfileProvider, Mặc định Provider lưu trữ dữ liệu Profile trong cơ sở dữ liệu MS SQL Server 2005 Express được đặt tên là ASPNETDB.mdf, được định vị trong thư mục App_Data. Nếu cơ sở dữ liệu không tồn tại thì nó sẽ được tạo tự động khi chạy chương trình có sử dụng Profile.

MẶc định bạn không thể lưu trữ thông tin Profile cho một người sử dụng nặc danh. ASP.NET Framework tính đồng nhất authenticate của bạn kết hợp với thông tin Profile, bạn có thể sử dụng đối tượng Profile với các kiểu chuẩn mà authentication hỗ trợ bởi ASP.NET Framework, bao gồm cả hai kiểm chứng Forms và Windows

**Creating Profile Groups**

Nếu bạn cần định nghĩa nhiều thuộc tính của Profile, bạn có thể tạo các thuộc tính bằng quản lý bởi việc tổ chức các thuộc tính trong Groups. Ví dụ trong file web.config sau định nghĩa hai nhóm thuộc tính Preferences và ContactInfo.

Ví dụ Trang web.config

```
<?xml version="1.0"?>

<configuration>
    <system.web>
      <profile>
        <properties>
          <group name="Preferences">
            <add name="BackColor" defaultValue="lightblue"/>
            <add name="font" defaultValue="Arial"/>
          </group>
          <group name="ContactInfo">
            <add name="Email" defaultValue="hiepgia@hiepgia.com"/>
            <add name="phone" defaultValue="0933030411"/>
          </group>
        </properties>
      </profile>
    </system.web>
</configuration>
```

Ví dụ sau đây sẽ hướng dẫn bạn cách tạo

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="showProfilegoups.aspx.cs" Inherits="showProfilegoups" %>
<%@ Import Namespace="System.Drawing" %>
<script runat="server">
    protected void Page_Load()
    {
        lblEmail.Text = Profile.ContactInfo.Email;
        lblPhone.Text = Profile.ContactInfo.phone;

        Style pagestyle = new Style();
        pagestyle.BackColor =
ColorTranslator.FromHtml(Profile.Preferences.BackColor);
        pagestyle.Font.Name = Profile.Preferences.font;
        Header.StyleSheet.CreateStyleRule(pagestyle, null,"html");
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

```
    <form id="form1" runat="server">
    <div>
        Email: <asp:Label ID="lblEmail" runat="server" /><br />
        Phone: <asp:Label ID="lblPhone" runat="server" />
    </div>
    </form>
</body>
</html>
```

**Hỗ trợ người sử dụng nặc danh**

Mặc định người sử dụng nặc danh không thể chỉnh sửa các thuộc tính của Profile, vấn đề là ASPNET Framework không có phương thức kết hợp dữ liệu Profile với người sử dụng riêng biệt trừ khi người sử dụng được kiểm chứng.

Nếu bạn muốn cho phép người sử dụng nặc danh chỉnh sửa các thuộc tính Profile, bạn có phải cho phép đặc tính của ASP.NET Framework được gọi là định danh nặc danh. Khi định danh nặc danh được cho phép, khi định danh duy nhất được gán đến người sử dụng nặc danh và được lưu trữ trong trình duyệt cookie ổn định.

Hơn thế nữa, bạn phải đánh dấu tất cả các thuộc tính Profile mà bạn muốn cho phép người sử dụng nặc danh với các đặc tính cho phép nặc danh. Ví dụ trang web.config sau cho phép định danh nặc danh và định nghĩa một thuộc tính Profile mà có thể chỉnh sửa được bởi người sử dụng nặc danh.

```
<?xml version="1.0"?>
<configuration>
    <system.web>
        <authentication mode="Windows" />
      <anonymousIdentification enabled="true"/>
      <profile>
        <properties>
          <add name="NumberOfVisits" type="Int32" defaultValue="0"
allowAnonymous="true"/>
        </properties>
      </profile>
    </system.web>
</configuration>
```

thuộc tính NumberOfVisits bao gồm thuộc tính allowAnonymous. Chú ý rằng file web.config và chỉ cho phép Form Authencation. Khi Form Authentication được cho phép và bạn không login, và khi đó bạn là người sử dụng nặc danh.

Trong ví dụ sau sẽ hướng dẫn cách bạn sửa thuộc tính định danh khi định danh nặc danh được cho phép.

Trang ShowAnonymousIdentification.aspx;

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

<script runat="server">
    void Page_PreRender()
    {
        lblName.Text = Profile.UserName;
        lblanonymous.Text = Profile.IsAnonymous.ToString();
```

```
        Profile.NumberOfVisits++;
        lblnumbetofanonymous.Text = Profile.NumberOfVisits.ToString();
    }

    protected void btnLogin_Click(object sender, EventArgs e)
    {
        FormsAuthentication.SetAuthCookie("Bob", false);
        Response.Redirect(Request.Path);
    }

    protected void btnLogout_Click(object sender, EventArgs e)
    {
        FormsAuthentication.SignOut();
        Response.Redirect(Request.Path);
    }
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Show Anonymous Identification</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        UseName: <asp:Label ID="lblName" runat="server" /><br />
        Is Anonymous: <asp:Label ID="lblanonymous" runat="server" />
        Number of Visits: <asp:Label ID="lblnumbetofanonymous"
runat="server" /><br /><hr />
        <asp:Button ID="btnReload" Text="Reload" runat="server" />
        <asp:Button ID="btnLogin" Text="Login" OnClick="btnLogin_Click"
runat="server" />
        <asp:Button ID="btnLogout" Text="Logout"
OnClick="btnLogout_Click" runat="server" />
    </div>
    </form>
</body>
</html>
```