

**ASP.NET**

GV: ThS Phạm Thị Vương

## Lợi ích của việc sử dụng ASP

- ❖ Công nghệ Server-side scripting
- ❖ Tự động biên dịch
- ❖ Cho phép tạo các ứng dụng web đơn giản nhanh chóng và dễ dàng
- ❖ Tạo trang web có tích hợp nội dung động

## Bất lợi của ASP

- ❖ Chỉ hỗ trợ 2 loại ngôn ngữ: VBScript và JavaScript
- ❖ Thông dịch mã lệnh ASP
- ❖ Pha trộn code, HTML và text
- ❖ Tương thích trình duyệt
- ❖ Không quản lý trạng thái trang web
- ❖ Cơ chế debug kém
- ❖ Tái sử dụng code kém

**Các điểm nổi bật của ASP.NET**

- ❖ Hỗ trợ đa ngôn ngữ: C#, VB.NET,..
- ❖ Biên dịch các trang trước, giúp làm tăng tốc độ thực hiện
- ❖ ASP code được phân ra độc lập với HTML và text
- ❖ Quản lý trạng thái trang web
- ❖ Có cơ chế hỗ trợ debug
- ❖ Hỗ trợ tái sử dụng code thông qua cơ chế kế thừa

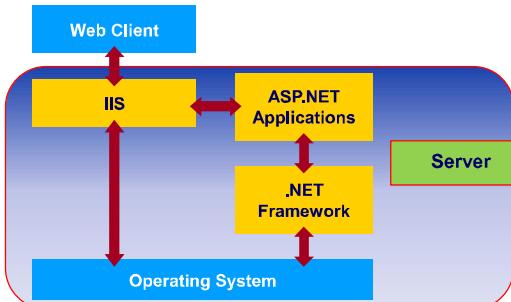
## Các điểm nổi bật của ASP.NET

- ❖ Dễ dàng sử dụng C# hoặc VB.NET
- ❖ Sử dụng cơ chế server-side caching
- ❖ Tự động nhận dạng trình duyệt người dùng đang sử dụng
- ❖ Đi cùng với nhiều server control được xây dựng sẵn
- ❖ Global.asax hỗ trợ nhiều sự kiện hơn
- ❖ Web service : triển khai một hàm từ xa thông qua web

## Giới thiệu về .NET Framework

- ❖ .NET Framework là nền tảng cho Microsoft.NET Platform
- ❖ .NET Framework là môi trường cho việc xây dựng, triển khai và vận hành các ứng dụng Web cũng như Web Service
- ❖ .NET Framework chứa Common Language Runtime (CLR) và các lớp thư viện cung cấp các dịch vụ cơ sở để xây dựng các ứng dụng

## Cấu trúc một ứng dụng ASP.NET

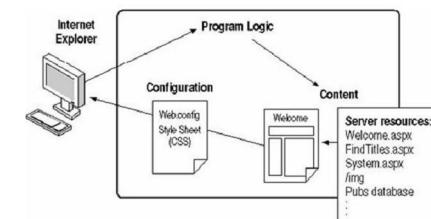


## Các thành phần của một ứng dụng Web

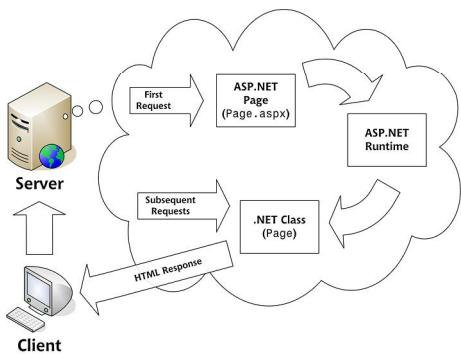
- ❖ **Các file hiển thị thông tin**
  - Web Forms ( .aspx ), HTML, images, audio, video, ...
- ❖ **Special Folder:**
  - App\_Code, App\_Data, App\_Themes, ...
- ❖ **Source code xử lý logic của chương trình**
  - .cs , .vb, ...
- ❖ **Các file cấu hình**
  - Web.config , Style sheets ( css )

## Các thành phần của một ứng dụng Web

- ❖ Trong một ứng dụng Web hoàn chỉnh, các phần thực thi của Web Form được lưu trong các file .dll và chạy trên server thông qua điều khiển của IIS



## Cơ chế xử lý một trang ASP.NET



## Cơ chế xử lý một trang ASP.NET

- ❖ **Các bước xử lý**
  - Khi client request một trang ASP.NET từ trình duyệt
  - Một HTTP request được gửi tới IIS trên Server
  - Web Server sẽ chuyển request này đến Asp.net runtime
  - Asp.net runtime chịu trách nhiệm tìm và load nội dung trang aspx được yêu cầu và biên dịch nó thành 1 lớp .NET để xử lý request
  - Lớp này sau đó sẽ phát sinh nội dung mã HTML trả về trình duyệt của người dùng

## Cơ chế xử lý một trang ASP.NET

- ❖ Người dùng thực hiện các thao tác trên trang web được trả về. Nếu các thao tác này đòi hỏi các xử lý tại server, thì trang này sẽ được gửi lại (post-back) về server. Thông tin trả về chứa các control ẩn chứa các thông tin về thao tác thực hiện của người dùng trên trang.
- ❖ Tại server, trang aspx được load lại, nhưng chỉ các trường ẩn mới được đọc và các sự kiện tương ứng mới được xử lý.
- ❖ Kết quả lại được gửi lại về browser.

## Cấu trúc một Web Form aspx

❖ Một Web Form bao gồm các thành phần:

- Directives
- Code Declaration Blocks
- Code Render Blocks
- Web Controls
- Server-side comments
- Literal Text và HTML Tags

Code Declaration Blocks và Code Render Blocks có thể đặt trực tiếp trên WebForm hoặc tách biệt trong file Code Behind

Company Logo

## Cấu trúc một Web Form aspx

```

<%@ Page Language="C#" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Sample Page</title>
</head>
<body>
    void Page_Load()
    {
        messageLabel.Text = "Hello World";
    }
</script>
<!-- comment here -->
<form runat="server">
    <p>
        <asp:Label id="messageLabel" runat="server" />
    </p>
    <p>
        <%-- Declare the title as string and set it --%>
        <% string Title = "This is generated by a code render " +
        "block." : %>
        <%= Title %>
    </p>
</form>
</body>
</html>

```

## Code declaration blocks

❖ Được khai báo nếu phần xử lý logic của chương trình được thể hiện ngay trong Web Form ( không sử dụng code behind )

❖ Khai báo các phương thức hoặc các hàm xử lý sự kiện

Ví dụ:  
`<script runat="server">
 void mySub()
 {
 // Code here
 }
 void Page_Load( ... )
 {
 // Code here
 }
</script>`

## Code Render Blocks

❖ Là các đoạn code được thực thi khi một trang được nạp hoặc trả nội dung về phía người dùng.

❖ Bao gồm 2 loại:

- Inline Code
- Inline Expression

## Directives

- ❖ Chứa các chỉ thị cho biết cách thức Web Form được biên dịch
- ❖ Được khai báo trong <%@ ... %> và có thể đặt tại bất kỳ vị trí nào trên Web Form
- ❖ Một số thuộc tính quan trọng: Language, AutoEventWireup, CodeFile, ...

Ví dụ:

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

## Code Render Blocks - Inline Code

❖ Bao gồm các lệnh xử lý trên server nhưng không trả nội dung về phía trình duyệt.

❖ Thường được sử dụng để khai báo biến

❖ Được khai báo trong cặp thẻ <% ... %>

Ví dụ:

`<%
string Title = "This is generated by a code render
block.";
%>`

## Code Render Blocks - Inline Expression

- ❖ Code xử lý trả thông tin về trình duyệt.
  - ❖ Thông tin trả về có thể là nội dung của một biến hoặc kết quả của việc gọi thực hiện một phương thức
  - ❖ Được khai báo trong cặp thẻ <%= ... %>
- Ví dụ:
- ```
<%
string Title = "This is generated by a code render block.";
%>
<%= Title %> hoặc <%= mySub() %>
```

## Web Controls

- ❖ Bao gồm 3 loại:
  - Html Control
  - Html Server Control
  - Asp.net Server Control
- ❖ Được khai báo trong thẻ
 

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

Ví dụ:

```
<asp:Label ID="Label1" runat="server" Text="Text Content" />
<asp:TextBox ID="TextBox1" runat="server" Text="Enter text here"
/>
```

## Server-side comments

- ❖ Thể hiện các ghi chú trên Web Form
- ❖ Sử dụng một trong 2 dạng:
  - Html Comment : <!-- comment -->
  - Asp.net Comment: <%-- comment --%>
- ❖ Html comment sẽ được gởi về trình duyệt do đó không thích hợp để comment nội dung Asp.net server-side code
- ❖ Html comment được dùng để ẩn thông tin đối với trình duyệt nhưng sẽ được xử lý bởi Asp.net runtime

## Server-side comments

- ❖ Ví dụ:
 

```
<!--
<% string Title = "This is generated by a code render block.";
%>
<%= Title %>
-->
```
- Kết xuất tại trình duyệt:
- ```
<!--
This is generated by a code render block.
-->
```

## Literal Text và HTML Tags

- ❖ Cung cấp cấu trúc định dạng thông tin trang web ( thông qua các thẻ Html ) cùng với nội dung hiển thị tĩnh ( literal text )
- ❖ Nếu không có thành phần này, trang web sẽ không có cấu trúc và trình duyệt sẽ không hiển thị được

## Literal Text và HTML Tags

Ví dụ:

```
<%@ Page Language="C#" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Sample Page</title>
<script runat="server">
void Page_Load()
{
    messageLabel.Text = "Hello World";
}</script>
</head>
<body>
<form runat="server">
<p>
<asp:Label id="messageLabel" runat="server" />
</p>
<p>
<%-- Declare the title as string and set it --%>
<% string Title = "This is generated by a code render block.";%>
<%= Title %>
</p>
</form>
</body>
</html>
```

## Web Form được xây dựng với code behind

- ❖ **Code behind:** là file mã nguồn ( C#, VB.net ) chứa khai báo lớp có nhiệm vụ xử lý các logic nghiệp vụ của chương trình hay các sự kiện xảy ra khi người dùng tương tác với WebForm
- ❖ **Tên của lớp thường trùng với tên của WebForm**  
Ví dụ: nếu tên WebForm là index → tên class sẽ là index
- ❖ **Tất cả các class xử lý sự kiện trên WebForm đều kế thừa từ lớp System.Web.UI.Page**
- ❖ **Tất cả các class đều chứa hàm Page\_Load tự động gọi thực hiện khi WebForm nhận được request từ client**
- ❖ **Hàm Page\_Load dùng để khởi tạo nội dung của các control trên WebForm**

## Web Form được xây dựng với code behind

### Ví dụ:

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

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (Page.IsPostBack == false)
        {
            TextBox1.Text = "initialized data";
        }
    }
}
```

## LÀM VIỆC VỚI CONTROL

Company Logo

- ❖ **Server Control**
- ❖ **ASP.NET Server control vs HTML control**
- ❖ **Simple control**
  - Label, Button (Button, LinkButton, ImageButton), TextBox
  - List Control (ListBox, DropDownList, Table, DataGrid, DataList, Repeater)
- ❖ **Validation control**
- ❖ **Một số control khác**

## Server Control

- ❖ **Server control là những control mà Web server (IIS) có thể “hiểu được”.**
- ❖ **Các loại server control**
  - HTML Server Control
  - ASP.NET Server Control
- ❖ **Dùng để thể hiện giao diện web**

Company Logo

Company Logo

## HTML Server Control

- ❖ **HTML Server control là những tag do HTML tạo ra**
- ❖ **Duy trì tương thích với các tag HTML cũ**
- ❖ **Thêm vào thuộc tính run at = “server”**
- ❖ **Tất cả HTML Server Control phải được đặt trong tag <form> với thuộc tính run at = “server”**

Company Logo

## ASP.NET Server Control

- ❖ ASP.NET Server Control là những tag đặc biệt của riêng ASP.NET
- ❖ Các control này cũng sẽ được xử lý trên server, và đòi hỏi phải có thuộc tính runat = "server"
- ❖ Không tương ứng với HTML tag nào
- ❖ Có thể dùng thẻ hiện các thành phần phức tạp

Company Logo

## Khác biệt trong HTML tag

- ❖ **Server control:**
  - <asp:controlnameid="some\_id" runat="server"/>
- ❖ **HTML control**
  - HTML tag

```
<asp:TextBox id="txtText" runat="server"></asp:TextBox>
<INPUT type="text" id="textfield1">

<asp:Button id="btnShow" runat="server"
Text="Show"></asp:Button>
<INPUT type="button" value="Show">
```

Company Logo

## ASP.NET Server control vs HTML control

Tính năng	ASP.NET Server control	HTML control
Server event	Kích hoạt được một số sự kiện cụ thể trên Server	Chỉ có thể kích hoạt các sự kiện mức trang trên server (post-back)
Quản lý trạng thái	Đữ liệu nhập vào control được lưu giữ lại sau mỗi request	Đữ liệu không được lưu giữ lại, phải tự lưu và diễn vào sử dụng script
Tương thích	Tự động nhận diện loại trình duyệt và tạo hiển thị cho phù hợp	Không tự động nhận diện trình duyệt
Các thuộc tính	.NET Framework cung cấp một tập các thuộc tính cho mỗi control, cho phép thay đổi phản hồi hiển thị và hành vi thông qua mã lệnh	Chỉ có các thuộc tính chuẩn của HTML

Company Logo

## Tại sao sử dụng HTML control ?

- ❖ **Sử dụng HTML control khi**

- Nâng cấp từ ASP
- Không phải tất cả các control đều cần các sự kiện server-side hoặc quản lý trạng thái

Company Logo

## Server Control & HTML Control

	Server control	HTML control
Hiển thị Text	Label, TextBox, Literal	Label, Text Field, Text Area, Password Field
Hiển thị Table	Table, DataGrid	Table
List	DropDownList, ListBox, DataList, Repeater	List Box, Dropdown
Thực hiện lệnh	Button, LinkButton, ImageButton	Button, Reset Button, Submit Button
Đặt giá trị	CheckBox, CheckBoxList, RadioButton, RadioButtonList	Checkbox, Radio Button
Hiển thị Image	Image, ImageButton	Image
Liên kết	Hyperlink	Anchor <a>

Company Logo

## Server Control & HTML Control

Group control	Panel, Placeholder	Flow Layout, Grid Layout
Ngày tháng	Calendar	none
Quảng cáo	AdRotator	none
Đường kẻ	none	Horizontal Rule
Lấy tên file từ client	none	File Field
Lưu dữ liệu trên trang	(sử dụng quản lý trạng thái)	Input Hidden
Kiểm tra tính đúng đắn của dữ liệu nhập	RequiredFieldValidator, CompareValidator, RangeValidator, RegularExpressionValidator, CustomValidator, ValidationSummary	none (sử dụng client script)

Company Logo

## Label, Buttons, TextBox - HTML tag

### ■ Label

- `<asp:Label id="Label1" runat="server">Please input text</asp:Label>`

### ■ Buttons (Button, LinkButton, ImageButton)

- `<asp:Button id="Button1" runat="server" Text="Button"></asp:Button>`

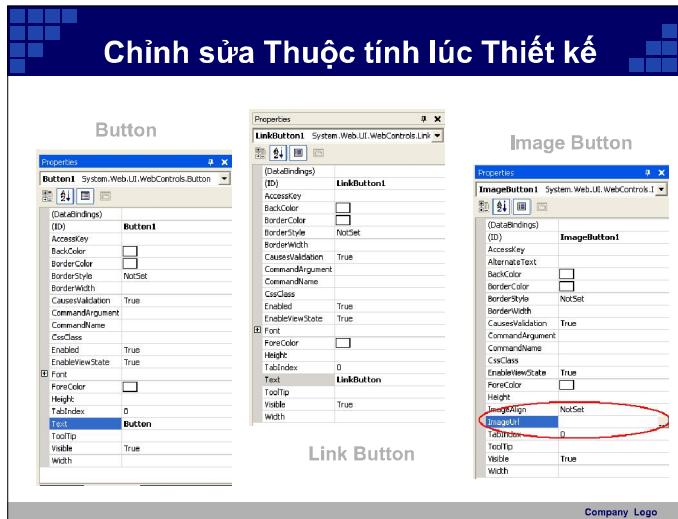
- `<asp:LinkButton id="LinkButton1" runat="server">LinkButton</asp:LinkButton>`

- `<asp:ImageButton id="ImageButton1" runat="server"></asp:ImageButton>`

### ■ TextBox

- `<asp:TextBox id="TextBox1" runat="server"></asp:TextBox>`

Company Logo



Company Logo

## List Control ListBox, DropDownList, Table

Control	Sử dụng khi
ListBox	Hiển thị danh sách dữ liệu read-only đơn giản, sử dụng scroll
DropDownList	Hiển thị danh sách dữ liệu read-only đơn giản, sử dụng cursor sò sò xuống
Table	Hiển thị thông tin dưới dạng dòng và cột. Table control cho phép xây dựng các bảng động bằng mã lệnh sử dụng các thuộc tính tập hợp TableRows và TableCells

## Chỉnh sửa Thuộc tính lúc Thiết kế

## Một số thuộc tính quan trọng

### ■ Label, Buttons

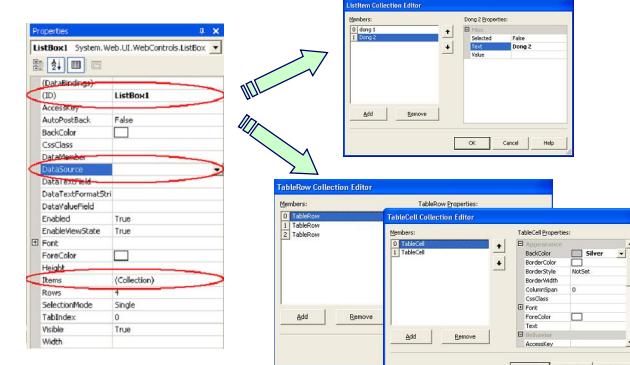
- Thuộc tính Text

### ■ TextBox

Thuộc tính	Sử dụng để
Text	Lấy/Đặt dữ liệu cho TextBox.
TextMode	SingleLine, MultiLine (scrollable), Hoặc Password.
Enabled	Enable/Disable TextBox
Visible	Show/Hide TextBox
ReadOnly	Ngăn không cho người dùng thay đổi dữ liệu trong TextBox.
AutoPostBack	Khi được thiết lập là True, sự kiện TextChanged trong TextBox sẽ tự động kích hoạt post-back về server (không phải được kích hoạt khi nội dung TextBox thay đổi)

Company Logo

## Chỉnh sửa Thuộc tính lúc Thiết kế



## Thêm các mục dữ liệu vào thời điểm chạy ứng dụng

- ❖ **ListBox và DropDownList:**
    - Sử dụng phương thức Add và danh sách Items của control
    - Ví dụ:
- ```
protected void btnShow_Click(object sender, EventArgs e)
{
    ListBox1.Items.Add(txtSource.Text);
    DropDownList1.Items.Add(txtSource.Text);
}
```

## Lấy mục dữ liệu được chọn

- ❖ Dùng thuộc tính **SelectedItem** để lấy mục dữ liệu được chọn hiện tại trong List

```
protected void Page_Load(object sender, EventArgs e)
{
    // Test if there is a selected item.
    if (ListBox1.SelectedItem != null)
        // Display the selected item.
        Label1.Text = "The selected item is: " +
                      ListBox1.SelectedItem.Text;
    else
        Label1.Text = "No item is selected.";
}
```

## Một số control khác

- ❖ **Lấy và thiết lập giá trị**
  - RadioButton, RadioButtonList, CheckBox, CheckBoxList
- ❖ **Gom nhóm**
  - Panel
- ❖ **Hiển thị Hình ảnh và Quảng cáo**
  - Background, Foreground, Image, AdRotator
- ❖ **Lấy thông tin Ngày tháng**
  - Calendar
- ❖ **Lấy Tập tin từ Client**
  - File Field HTML control

## Một số control khác RadioButton, CheckBox

- ❖ **HTML tag**
  - <asp:RadioButton id="R1" runat="server"></asp:RadioButton>
  - <asp:CheckBox id="C1" runat="server"></asp:CheckBox>
- ❖ **Sử dụng thuộc tính GroupName để thiết lập giá trị thiết lập**

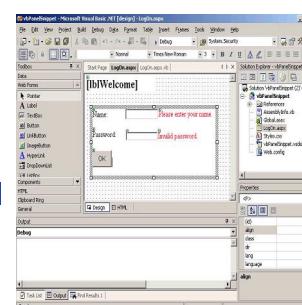

```
protected void
{
    if (CheckBox1.Checked)
        Response.Redirect("http://www.google.com");
}
```

  - Tất cả RadioButton phải có cùng một GroupName

## Panel

- ❖ **Kéo thả control Panel vào Web form.**

- ❖ **Kéo các control khác lên trên Panel để gom nhóm**



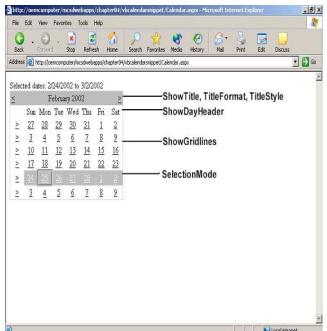
## Hình ảnh và Quảng cáo

- ❖ **Hình nền**
  - Sử dụng thuộc tính **Background** của Web form
  - Sử dụng thuộc tính **BackImageUrl** của Panel control
- ❖ **Hình ảnh**
  - Sử dụng **Image control**
- ❖ **Button bằng hình ảnh**
  - Sử dụng **ImageButton control**
- ❖ **Quảng cáo**
  - Sử dụng **AdRotator control**

## Calendar

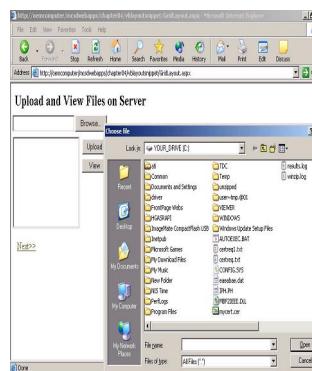
- ❖ Sử dụng Calendar control để lấy thông tin về Ngày tháng
- ❖ Để lấy hoặc thiết lập giá trị ngày tháng trên Calendar control, sử dụng hàm xử lý sự kiện

SelectionChanged và thuộc tính SelectedDate hoặc SelectedDates



## File Field HTML control

- ❖ Sử dụng File Field HTML control để upload file từ client lên server
- ❖ File Field HTML control = Text Field HTML control + Submit Button HTML control
- ❖ Nhấn vào Browse button sẽ hiển thị cửa sổ cho phép chọn đường dẫn đến các file muốn upload trên máy client



## Sử dụng Css đối với các asp.net controls

### Cách 2:

- Thay đổi nội dung thuộc tính Style ( inline css ) của đối tượng server control tương ứng

Ví dụ:

```
<asp:label id="labMsg" runat="server" Text="hello world"
style="font-style: italic; text-decoration: underline; color: Red" />
Code behind:
labMsg.Style["font-style"] = "italic";
labMsg.Style["text-decoration"] = "underline";
labMsg.Style["color"] = "Red";
```

## Sử dụng Css đối với các asp.net controls

### Cách 3:

- Thiết lập giá trị thuộc tính CssClass của đối tượng server control tương ứng ( embedded , external css )

Ví dụ:

```
.myStyle {
    font-style: bold;
    text-decoration: line-through;
    color: green; }
```

Code behind:

```
labMsg.Style.Clear();
labMsg.CssClass = "myStyle";
```

## Sử dụng Css đối với các asp.net controls

### Cách 1:

- Khai báo trực tiếp trong thuộc tính style ( inline css ) của thẻ asp.net controls

Ví dụ:

```
<asp:label id="labMsg" runat="server" Text="hello world"
style="font-style: italic; text-decoration: underline; color: Red" />
```

- Sử dụng các thuộc tính định dạng built-in được hỗ trợ đối với asp.net controls tương ứng

```
<asp:label id="labMsg" runat="server" ForeColor="Blue" />
```

## Theme, Skin

- ❖ Theme, skins cho phép xây dựng các tập kiểu định dạng hiển thị ( style ) có thể áp dụng đối với mọi asp.net server controls trong toàn bộ site

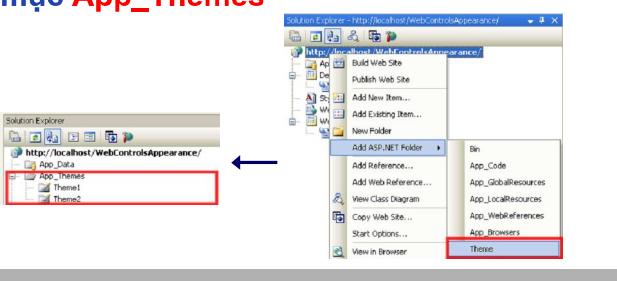
- ❖ Có thể xây dựng các style áp dụng đối với các control đơn giản ( Label, TextBox... ) và phức tạp ( GridView .. ) mà css không áp dụng được

- ❖ Mỗi theme có thể xem như 1 giao diện của trang web

- ❖ Giúp tạo ra giao diện nhất quán cho toàn bộ trang web

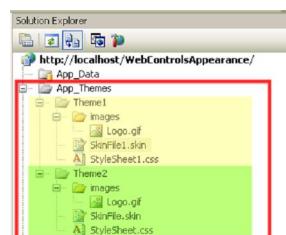
## Theme

- ❖ Một asp.net website có thể định nghĩa nhiều theme.
- ❖ Tất cả các theme phải đặt trong thư mục **App\_Themes**



## Theme

- ❖ Mỗi theme có thể định nghĩa nhiều **skin** file, **css** file, hình ảnh, ...



## Skin

- ❖ Skin file: mô tả tập các kiểu định dạng hiển thị của các asp.net server controls

### Ví dụ:

Định nghĩa style áp dụng đối với mọi Label và TextBox trong trang web

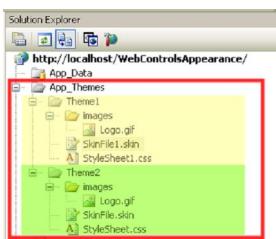
```
<asp:Label runat="server" ForeColor="Blue" Font-Size="10pt" Font-Name="Verdana" />
<asp:TextBox runat="server" BackColor="#FFFFC0" ForeColor="Green" />
```

Các định dạng trên không chứa thuộc tính ID hay bất kỳ thuộc tính nào liên quan đến một control cụ thể (Ví dụ: Text ... )

## Theme, Skin

- ❖ Sử dụng Theme trong WebForm

```
<%@ Page ... Theme="Theme1" ...>
```



## Theme, Skin

- ❖ Sử dụng Theme đối với mọi WebForm trong site

Trong file **web.config**, bổ sung:

```
<system.web>
  ...
  <pages theme="Theme1" />
  ...
</system.web>
```

→ Tất cả các WebForm khi thêm mới vào site đều sử dụng **Theme1** như là theme mặc định. Nếu WebForm này khai báo thuộc tính **Theme** → Theme trong WebForm sẽ được ưu tiên sử dụng

## Overriding Theme

- ❖ Giả sử trong **theme1** định nghĩa style của **TextBox**

```
<asp:TextBox runat="server" BackColor="#FFFFC0" ForeColor="Green" />
```

- ❖ Trong WebForm định nghĩa 1 **TextBox** với định dạng

```
<asp:TextBox runat="server" BackColor="#FF8000" ForeColor="Fuchsia" ID="TextBox1" />
```

→ Style trong theme sẽ override style cụ thể của control trong WebForm

## Overriding Theme

- ❖ Để override style định nghĩa trong theme

### Cách 1:

```
<asp:TextBox ID="TextBox1" ... EnableTheming="false" ...
```

### Cách 2:

```
<%@ Page ... StyleSheetTheme="Theme1" ...
```

- style trong **Theme** sẽ được sử dụng sau khi áp dụng style cụ thể của server control trong WebForm  
 → style trong **StyleSheetTheme** sẽ được sử dụng trước khi áp dụng style cụ thể của server control trong WebForm

## Named skin

- ❖ Trong trường hợp một loại control cần thiết lập nhiều định dạng khác nhau, ta có thể phân biệt giữa các định dạng này thông qua thuộc tính **SkinID**

### Ví dụ:

```
<asp:TextBox runat="server" BackColor="#FFFFC0" ForeColor="Green" />  

<asp:TextBox runat="server" BackColor="#FF0000" ForeColor="White" SkinID="skin1" />
```

### Sử dụng trong WebForm:

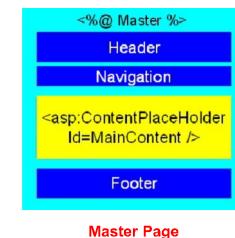
```
<asp:TextBox ID="TextBox1" runat="server" />  

<asp:TextBox ID="TextBox1" runat="server" SkinID="skin1" />
```

## Master Page

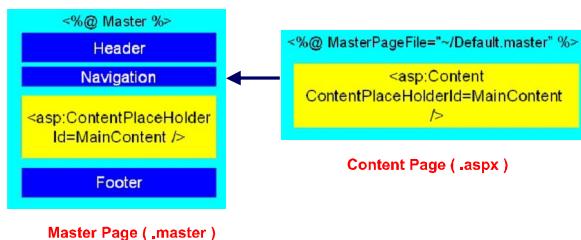
- ❖ **Master Page** cho phép định nghĩa layout template nhất quán cho các WebForm trong site

### Ví dụ:



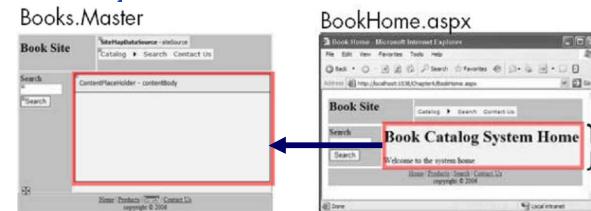
## Master Page

- ❖ **Content Page** là các WebForm kế thừa layout template mà Master Page đã định nghĩa và bổ sung thêm nội dung tương ứng với chức năng của WebForm này



## Master Page

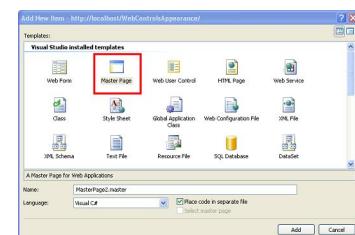
- ❖ Master page sẽ định nghĩa các **PlaceHolderControl**.
- ❖ Content page sẽ chèn nội dung tương ứng vào các PlaceHolderControl trong



## Master Page

### Tạo Master page.

Website → Add New Item → Master page

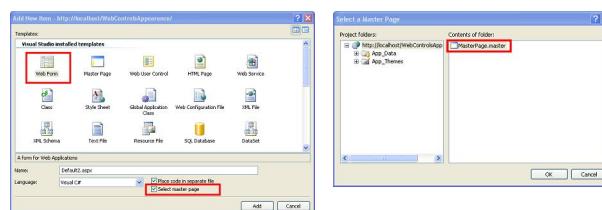


## Master Page

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs"
Inherits="MasterPage" %>
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:contentplaceholder id="ContentPlaceHolder1" runat="server"></asp:contentplaceholder>
        </div>
    </form>
</body>
</html>
```

## Master Page

### Tạo Content page



## Master Page

### Content page

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true"
CodeFile="Default2.aspx.cs"
Inherits="Default2" %>
<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">
</asp:Content>
```

**Master page**

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs"
Inherits="MasterPage" %>
...
<asp:contentplaceholder id="ContentPlaceHolder1" runat="server">
</asp:contentplaceholder>
...
```

## Truy xuất nội dung trong Master Page

❖ Trong một số trường hợp, ta cần truy xuất và thay đổi một số nội dung trong Master page từ Content Page.

❖ Cách 1:

Sử dụng phương thức **FindControl** từ đối tượng Master

Ví dụ:

```
HyperLink ad = (HyperLink)Master.FindControl("controlInMasterPage");
if (ad != null)
{
    ad.ImageUrl = "images/Logo.gif";
    ad.NavigateUrl = "http://www.interneturl.com";
}
```

## Truy xuất nội dung trong Master Page

❖ Cách 2:

Đóng gói dữ liệu trong Master page thành các thuộc tính có thể truy xuất ( Properties ).

Content page sẽ truy xuất dữ liệu của Master page thông qua các Properties này.

## Truy xuất nội dung trong Master Page

❖ Ví dụ: khai báo 2 properties trong master page

```
public partial class ProgrammedContentMaster : System.Web.UI.MasterPage
{
    public string AdImageUrl
    {
        get { return imgbtnAd.ImageUrl; }
        set { imgbtnAd.ImageUrl = value; }
    }
    public string AdNavigateUrl
    {
        get { return imgbtnAd.NavigateUrl; }
        set { imgbtnAd.NavigateUrl = value; }
    }
}
```

❖ Truy xuất từ Content Page

```
ProgrammedContentMaster pcm = (ProgrammedContentMaster)Master;
pcm.AdImageUrl = "~/Images/something.gif";
pcm.AdNavigateUrl = "http://www.somewhereelse.com";
```

## Truy xuất nội dung trong Master Page

### ❖ Cách 3: chỉ định kiểu cụ thể của Master Page trong content page

```
<%@ Page Language="C#" MasterPageFile("~/MasterPage.master" .... %>
```

```
<%@ MasterType VirtualPath "~/MasterPage.master" %>
```

```
hoặc
```

```
<%@ MasterType TypeName="MyMasterPageClassName" %>
```

### ❖ Theo cách này, việc truy xuất đến Master page từ content page trong code behind không cần thực hiện ép kiểu

```
this.Master.AdImageUrl = "~/Images/something.gif";
this.Master.AdNavigateUrl = "http://www.somewhereelse.com";mm42
1
```



## Validation Controls

### Sử dụng Validation Control

#### Các bước:

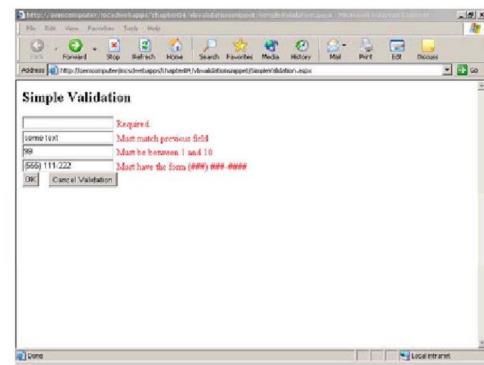
- Kéo thả 1 validate control vào Web form
- Thiết lập các thuộc tính cho validate control:
  - ControlToValidate là control bạn muốn kiểm tra
  - ErrorMessage : Thông báo lỗi
  - Text : Hiển thị của validate control
- Sử dụng ValidationSummary control để hiển thị tất cả các lỗi xảy ra trong trang

Mặc dù việc kiểm tra xảy ra ở client, nhưng nó chỉ thực hiện khi có 1 sự kiện post-back xảy ra!

Company Logo

## Ví dụ

### ▪ Đặt thuộc tính **ErrorMessage** cho Validate control



Company Logo

## Validation Controls

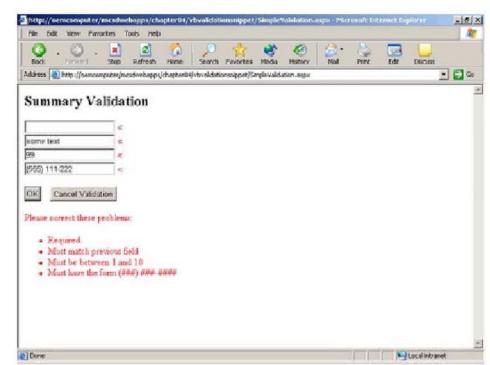
- Validation control kiểm tra tính đúng đắn của dữ liệu do người nhập vào trước khi trang được gửi về cho server

Validation control	Sử dụng khi
RequiredFieldValidator	Kiểm tra nếu dữ liệu trong control khác rỗng
CompareValidator	Kiểm tra nếu mục dữ liệu nhập trong control giống với control khác
RangeValidator	Kiểm tra nếu mục dữ liệu nhập trong control nằm trong khoảng 2 giá trị
RegularExpressionValidator	Kiểm tra nếu mục dữ liệu nhập trong control thỏa 1 công thức định dạng chỉ định
CustomValidator	Kiểm tra tính đúng đắn của dữ liệu nhập vào control sử dụng client-side script hoặc a server-side code, hoặc cả 2
ValidationSummary	Hiển thị tất cả các lỗi kiểm tra xảy ra trong trang

Company Logo

## Ví dụ

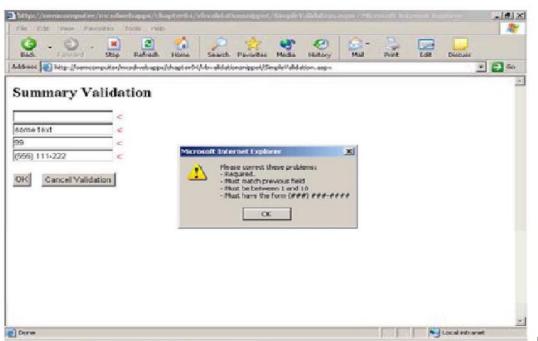
### ▪ Sử dụng ValidationSummary



Company Logo

## Ví dụ

- Sử dụng ValidationSummary control với ShowMessage=True



### Required Field Validator

Thuộc tính	Ý nghĩa
ControlToValidate	Control dùng để thực hiện kiểm tra
InitialValue	Giá trị dùng để so sánh. Mặc định là rỗng.

Mẫu số: <asp:TextBox ID="txtMauSo" runat="server"></asp:TextBox>

```
<asp:RequiredFieldValidator ID="reqMauSo" runat="server"
    ControlToValidate="txtMauSo"
    InitialValue="0"
    Text="Passwords must match" />
```

### Compare Validator

Thuộc tính	Ý nghĩa
ControlToValidate	Control dùng để thực hiện kiểm tra
ControlToCompare	Control dùng để so sánh
Operator	Toán tử so sánh: Equal, GreaterThan, GreaterThanEqual, LessThan, LessThanEqual, NotEqual, hoặc DataTypeCheck
ValueToCompare	Giá trị hằng số để so sánh ControlToValidate với Operator
Type	Kiểm tra kiểu dữ liệu của ControlToValidate: String, Integer, Double, Date, Currency

## Ví dụ - Kiểm tra Mật khẩu và Xác nhận mật khẩu

Enter Password:  Reenter Password:  Passwords must match

```
Enter Password: <asp:TextBox ID="txtPass1" runat="server" TextMode="password"></asp:TextBox><br>
Reenter Password: <asp:TextBox ID="txtPass2" runat="server" TextMode="password"></asp:TextBox>
<asp:CompareValidator ID="compPass" runat="server" ControlToValidate="txtPass2" Operator="Equal" ControlToCompare="txtPass1" Text="Passwords must match" />
```

Ví dụ - Kiểm tra dữ liệu là số nhỏ hơn hoặc bằng 18

Age:  You are too old to view this site

```
Age: <asp:TextBox ID="txtAge" runat="server"></asp:TextBox>
<asp:CompareValidator ID="compAge" runat="server" ControlToValidate="txtAge" ValueToCompare="18" Operator="LessThanEqual" Type="Integer" Text="You are too old to view this site" />
```

Email:  Enter a valid Email

```
Email <asp:TextBox ID="txtEmail" runat="server"></asp:TextBox>
<asp:RegularExpressionValidator ID="valEmail" runat="server" ControlToValidate="txtEmail" ValidationExpression="^([a-zA-Z0-9]+([._][a-zA-Z0-9]+)*@[a-zA-Z0-9]+([.-][a-zA-Z0-9]+)*\.[a-zA-Z]{2,4})$" ErrorMessage="Enter a valid Email" />
```

Regular Expression Validator	
Ký hiệu	Ý nghĩa
^ ... \$	Dấu hiệu bắt đầu và kết thúc một Expression
\t	Có chứa Ký tự Tab
\n	Có chứa Ký tự xuống dòng
.	Có chứa Ký tự bất kỳ khác \n
[qwerty]	Có chứa Ký tự bất kỳ trong ngoặc vuông
[^qwerty]	Không chứa ký tự nào trong ngoặc vuông
[a-zA-Z]	Có chứa ký tự trong khoảng từ a đến z
\w	Có chứa một từ bất kỳ (word). Tương tự [a-zA-Z0-9]
\W	Có chứa một chuỗi bất kỳ không phải là một từ (nonword)
	Hoặc

Company Logo

## Hủy bỏ việc Kiểm tra

- Để cho người dùng tự hủy bỏ việc kiểm tra nếu muốn
- Các bước:**
  - Tạo một HTML Button control

```
<INPUT id="butCancel" onclick="Page_ValidationActive=false;" type="submit" value="Cancel">
```

  - Hủy sự kiện validation: onclick="Page\_ValidationActive=false;"
  - Và gửi trang về cho server
  - Kiểm tra lại dữ liệu ở trên server
  - Kiểm tra thuộc tính Page.IsValid trong hàm xử lý sự kiện Page\_Load

Company Logo

## Page.IsValid

- Thuộc tính Page.IsValid kiểm tra xem các form đã thỏa các Validation Control hay không.
- Trả về true nếu tất cả đều được test thành công
- Trả về false, trong trường hợp ngược lại.

```
protected void Page_Load(object sender, EventArgs e)
{
    // Validate in case user cancelled validation.
    if (Page.IsPostBack == true)
        // Check if page is valid
        Page.Validate();

    else
        // User cancelled operation, return home
        Response.Redirect("default.aspx");
}
```

Company Logo

## CustomValidator

- Sử dụng CustomValidator control
- Tự viết mã lệnh kiểm tra chạy trên server hoặc client
- Trên Server
  - Đặt mã lệnh kiểm tra trong hàm xử lý sự kiện ServerValidate
- Hoặc Trên Client
  - Chỉ định đoạn script kiểm tra cho thuộc tính ClientValidationFunction của CustomValidator

Company Logo

## Ví dụ: Tự Kiểm tra trên Server

```
protected void MyValidate(object source, ServerValidateEventArgs args)
{
    args.IsValid = false;
    if (TextBox1.Text == "abc")
        args.IsValid = true;
}
```

Company Logo

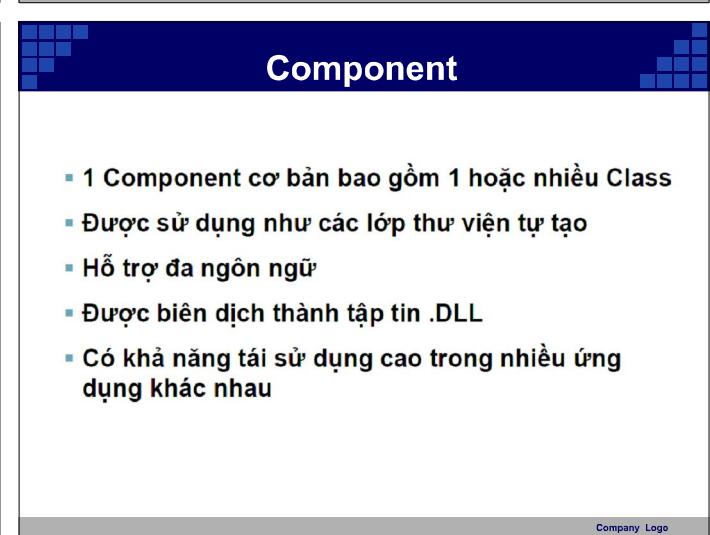
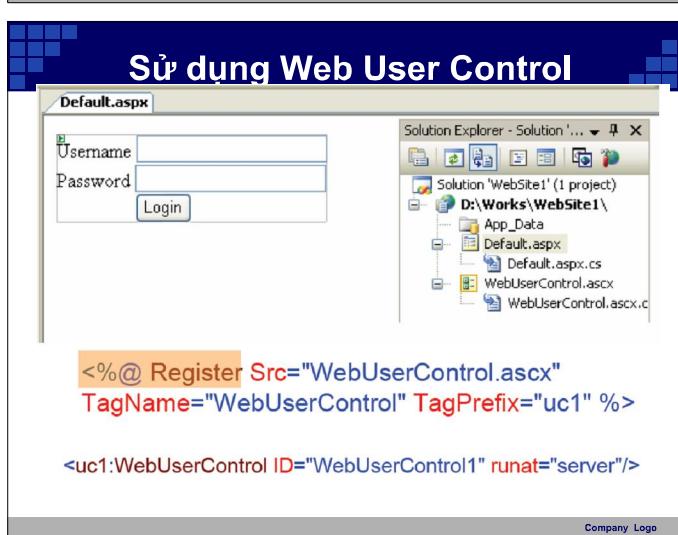
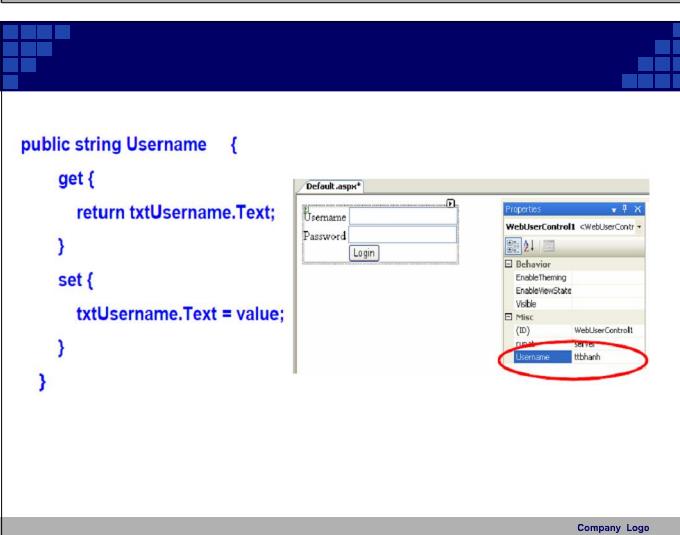
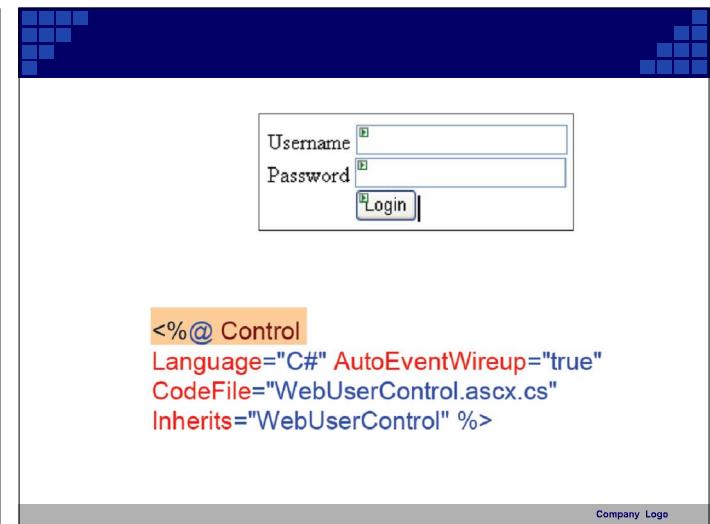
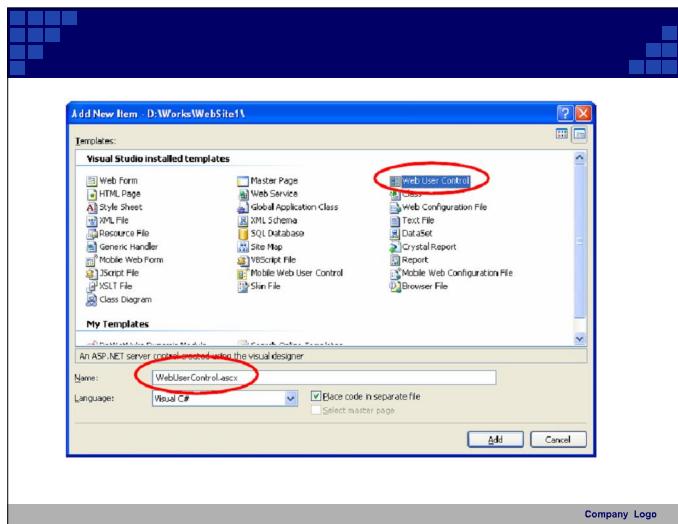
## Ví dụ: Tự kiểm tra trên Client

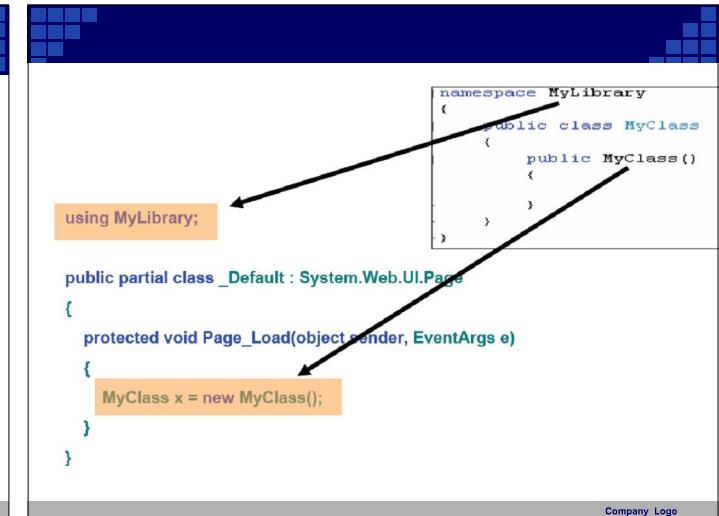
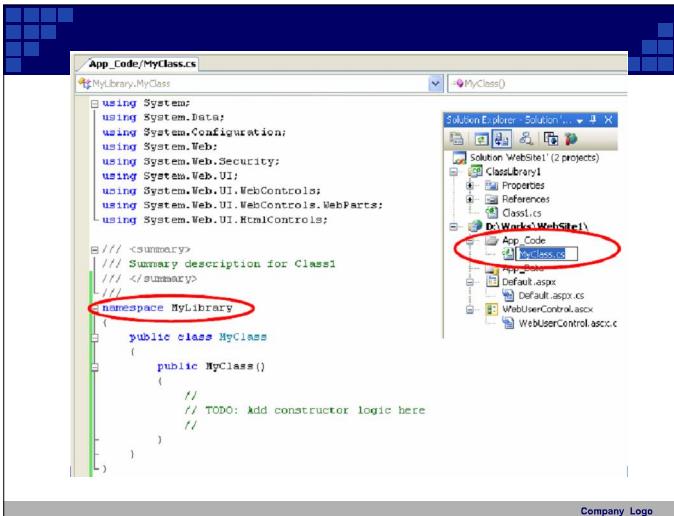
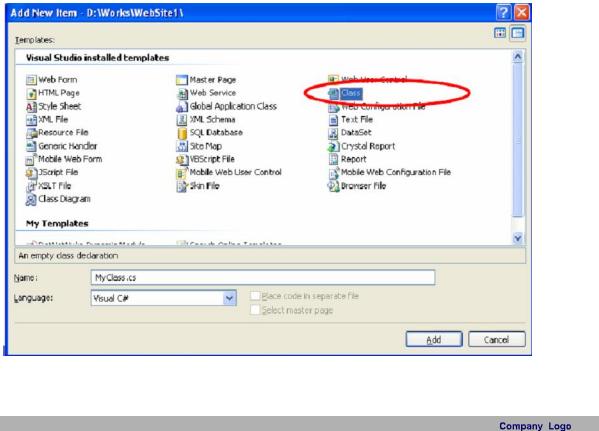
```
<script language="javascript">
function ClientValidate(e, args)
{
    args.IsValid = false;
    if (args.Value == "abc")
        args.IsValid = true;
}
</script>
```

Company Logo

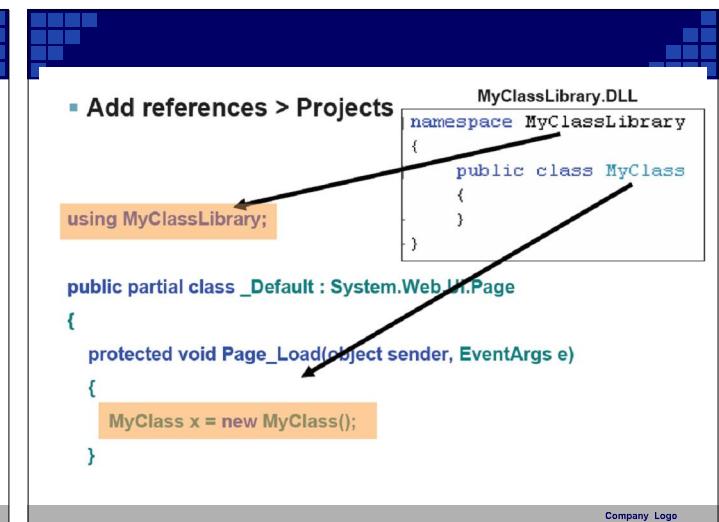
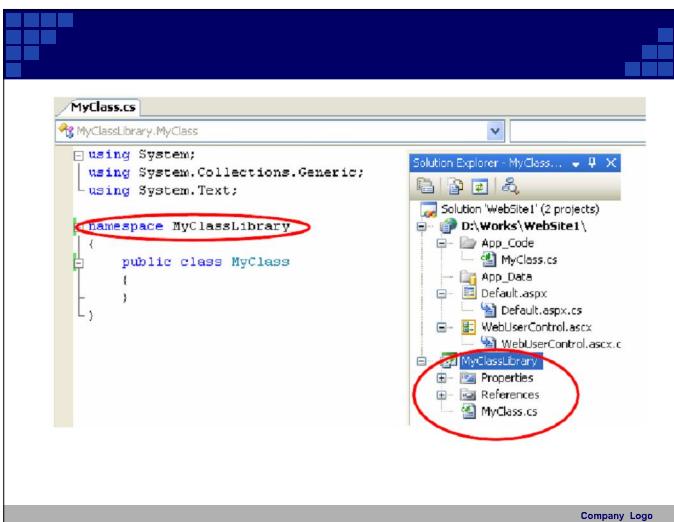
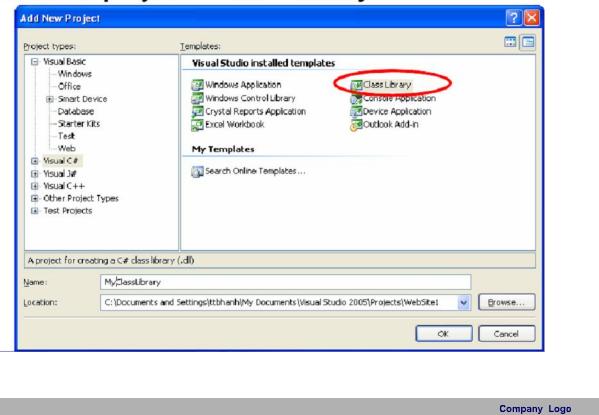
## Web UserControl

- Là các trang ASP.NET được sử dụng như là server control
- Có thể tự định nghĩa các thuộc tính & phương thức riêng
- Có khả năng tái sử dụng cho nhiều trang



**Cách 1****Cách 2**

## Add new project &gt; Class Library

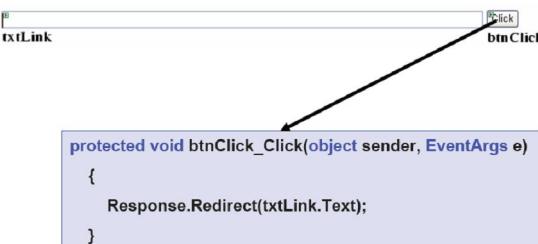


## Các đối tượng của ASP.NET

- Response
- Request
- Server
- Application
- Session

Company Logo

## Ví dụ



Company Logo

## Response Object

- Response là đối tượng được tạo ra tương ứng với mỗi yêu cầu của client
- Đối tượng Response thông thường dùng để xử lý các HTTP Request, và trả kết quả về cho client

### Thuộc tính

Charset      xác lập bộ charset sẽ truyền

IsClientConnected      cho biết hiện thời client có đang kết nối hay không

Cookies      các thông tin về Cookies sẽ được ghi xuống client

...

Company Logo

Phương thức	Mô tả
Write	ghi dữ liệu text
WriteFile	ghi dữ liệu từ file
BinaryWrite	ghi dữ liệu nhị phân
Close	đóng kết nối tới client
End	Kết thúc thi hành một trang
Redirect	chuyển client sang một URL khác
...	

Company Logo

## Request Object

- Cung cấp các thông tin về yêu cầu của client
- Được tạo ra tương ứng với các Http Request
- Dùng đối tượng này để đọc những thông tin client đã gửi (header, query string, cookies, ...)

Company Logo

Thuộc tính	Mô tả
Browser	Cung cấp thông tin liên quan đến trình duyệt của client
Url	Địa chỉ trang gửi request
Cookies	các thông tin về Cookies ở client sẽ được đọc lên
QueryString	Lấy tham số truyền từ client theo phương thức GET
Form	Lấy tham số truyền từ client theo phương thức POST
...	

Company Logo

## Ví dụ - Truyền tham số theo phương thức GET

- Cách tạo HTTP QueryString

- Gõ trực tiếp vào chuỗi URL

```
<A HREF="example.aspx?FirstName=Hanh&LastName=Tran"> string sample</A>
```

## Cookies

- Cookies là một mẩu thông tin nhỏ dùng để lưu trữ thông tin của người dùng trên máy tính.
- Cookies có thể được lưu trữ tạm thời hoặc lưu trữ lâu dài.
- Cookie lưu trữ tạm thời : sẽ không còn giá trị ngay khi người dùng rời khỏi web site.
- Cookie lưu trữ lâu dài : vẫn còn được lưu trữ trên máy của người dùng, và Web Server có thể đọc những thông tin này vào những lần kế tiếp người dùng vào web site.

## Sử dụng cookies

- Lệnh ghi cookies

- Response.Cookies[cookie][(key)].attribute = value;
  - cookie: tên biến
  - key : tham số tùy chọn, có thể đặt nhiều giá trị cho 1 cookie
  - attribute: thuộc tính (domain, path,...)

- Lấy giá trị cookies

- value = Request.Cookies[cookie][(key)].attribute

- Cookies đơn

- Ghi

```
Response.Cookies["userName"].Value = "mike";  
Response.Cookies["userName"].Expires = DateTime.Now.AddDays(1);
```

- Đọc

```
if (Request.Cookies["userName"] != null)  
    Label1.Text = Request.Cookies["userName"].Value;
```

- Thể hiện trong ASP.NET là lớp HttpCookie

- Các thuộc tính của HttpCookie

- Name : tên của Cookie
- Domain : domain cookie này thuộc về.
- Expires : xác định thời gian có hiệu lực của Cookie
- Value : Giá trị của Cookie
- HasKeys : Cookie có tập giá trị con hay không
- Values : tập các giá trị của Cookie

### Xóa Cookies

```
Response.Cookies["userName"].Expires = DateTime.Now.AddDays(-1);

Response.Cookies["userInfo"].Expires = DateTime.Now.AddDays(-1);
```

## Server Object

### Cung cấp các phương thức giúp

- Chuyển điều khiển giữa các trang với nhau
- Lấy các thông tin về mã lỗi, encode, ...

### Các thuộc tính

- MachineName : tên server
- ScriptTimeout : thời gian time-out của request

Phương thức	Mô tả
MapPath	Ánh xạ địa chỉ tương đối thành địa chỉ tuyệt đối trên server
HtmlEncode	Giữ nguyên tag HTML
HtmlDecode	Định dạng nội dung theo tag HTML
UriEncode	Mã hóa Url theo ASCII
UriDecode	Giải mã ASCII từ Url
Transfer	Lấy kết xuất từ trang khác rồi quay lại
Execute	Lấy kết xuất từ trang khác
...	

## Application Object

- Một ASP.NET application bao gồm tất cả các file, trang web, sự kiện, module và code trong phạm vi một thư mục web ảo (virtual directory) và các thư mục con của nó
- Đối với mỗi ASP.NET application, một Application Object được tạo ra để thể hiện tình trạng của ASP.NET application này
- Application Object được tạo khi client yêu cầu bất kỳ trang nào trong application này

### Application Object bị hủy khi

- Stop Web Server
- Server bị sập
- Hủy ASP.NET Application

### Sự kiện

- Application\_OnStart
- Application\_OnEnd
- (global.asax)

▪ Application Object chứa một danh sách các biến trạng thái dùng chung của application. Ta có thể sử dụng chúng để lưu trữ các thông tin xuyên suốt ứng dụng

▪ Các biến trạng thái này được lưu trữ thành từng cặp key-value.

- Key : tên trạng thái
- Value : giá trị trạng thái

```
Application["SoLan"] = 0;
```

```
//global.asax
<script language="C#" runat="server">
    void Application_OnStart(Object sender, EventArgs E) {
        Application["SoLan"] = 0;
    }
</script>

//Myform.aspx.cs
void Page_Load(Object Src, EventArgs E){
    Application["SoLan"] = (Int32) Application ["SoLan"] + 1;
    Response.Write("Số lần vào trang này :" + Application ["SoLan"]);
}
```

Company Logo

### Ví dụ – Đếm số lần duyệt 1 trang web (Cải tiến)

```
void Page_Load(Object Src, EventArgs E){
    Application.Lock();
    Application["SoLan"] = (Int32) Application ["SoLan"] + 1;
    Application.UnLock();
    Response.Write("Số lần vào trang này :" + Application ["SoLan"]);
}
```

Company Logo

## Session Object

- Một session (phiên làm việc) là một chuỗi các thao tác của người dùng trên cùng một web application
- Với mỗi phiên làm việc của client, sẽ có một Session Object được tạo ra
- Session Object sẽ cung cấp cho ta những thông tin về phiên làm việc hiện hành này của client

Company Logo

- Session được tạo ra khi client bắt đầu phiên làm việc của mình
- Session sẽ được hủy khi
  - Client tự động thoát khỏi session
  - Sau một khoảng thời gian (time-out), client không có hành động làm việc nào
- Sự kiện**
  - Session\_OnStart
  - Session\_OnEnd

(global.asax)

Company Logo

	Ý nghĩa
Session.Timeout = minutes	Đặt thời gian của phiên làm việc
Session.SessionID	ID của phiên làm việc
Session.Abandon()	Hủy phiên làm việc (hủy cả biến dữ liệu)

- Cung cấp cho ta một danh sách các trạng thái xuyên suốt session này
- Các trạng thái của Session Object cũng được tổ chức trong một Collections dưới dạng key-value

```
Session["DaDangNhap"] = 0;
```

Company Logo

### Ví dụ - Kiểm tra Quyền đăng nhập

- Làm thế nào để ngăn không cho người dùng truy cập vào các trang web nếu chưa đăng nhập?
- Ý tưởng**
  - Dùng các biến Session để lưu trạng thái đăng nhập của người dùng:
    - Session["IsLogin"] = 0/1 : Lưu trạng thái đăng nhập
    - Session["Username"] : Lưu Tên đăng nhập
    - Session["Authentication"] : Lưu Loại quyền đăng nhập
    - ...

Company Logo

Làm thế nào để ngăn không cho người dùng truy cập vào các trang web nếu chưa đăng nhập?

- Khởi tạo giá trị mặc định cho biến Session["IsLogin"] = 0 (chưa đăng nhập)

```
//global.asax
<script language="C#" runat="server">
    void Session_OnStart(Object sender, EventArgs E) {
        Session ["IsLogin"] = 0;
    }
</script>
```

Company Logo

Làm thế nào để ngăn không cho người dùng truy cập vào các trang web nếu chưa đăng nhập?

- Tạo trang Login.aspx cho phép người dùng đăng nhập
  - Nếu kiểm tra thông tin đăng nhập không đúng thì hiển thị thông báo yêu cầu đăng nhập lại.
  - Ngược lại, nếu ĐÚNG thì dùng một (hoặc nhiều) biến Session để lưu trạng thái login thành công lại.

```
//Login.aspx.cs
void btnLogin_Click(Object Src, EventArgs E){
    if (Thông tin đăng nhập đúng)
        Session ["IsLogin"] = 1;
    else
        Response.Write("Vui lòng nhập lại!");
}
```

Company Logo

Làm thế nào để ngăn không cho người dùng truy cập vào các trang web nếu chưa đăng nhập?

- Trong tất cả các trang muốn bảo mật, phải thêm đoạn mã sau để kiểm tra người dùng đã login hay chưa, nếu chưa thì redirect lại trang login.aspx

```
//MyForm.aspx.cs
void Page_Load(Object Src, EventArgs E){
    int nDaDangNhap = (Int32) Session ["IsLogin"];
    if (nDaDangNhap == 0)
        Response.Redirect("Login.aspx");
}
```

Company Logo

Làm thế nào để ngăn không cho người dùng truy cập vào các trang web nếu chưa đăng nhập?

- Tạo xử lý khi người dùng logout
  - Reset trạng thái login là chưa đăng nhập

```
//MyForm.aspx.cs
void btnLogout_Click(Object Src, EventArgs E){
    Session ["IsLogin"] = 0;
    Response.Redirect("Login.aspx");
}
```

Company Logo

## Các Tập tin trong một Ứng dụng Web

- Khi xây dựng một Ứng dụng Web:
  - Visual Studio .NET biên dịch tất cả **mã nguồn** vào một file .DLL lưu trong thư mục /bin
  - Phản **giao diện** của ứng dụng nằm ở các file .aspx và .html

Company Logo

### Chu trình sống của một ứng dụng web

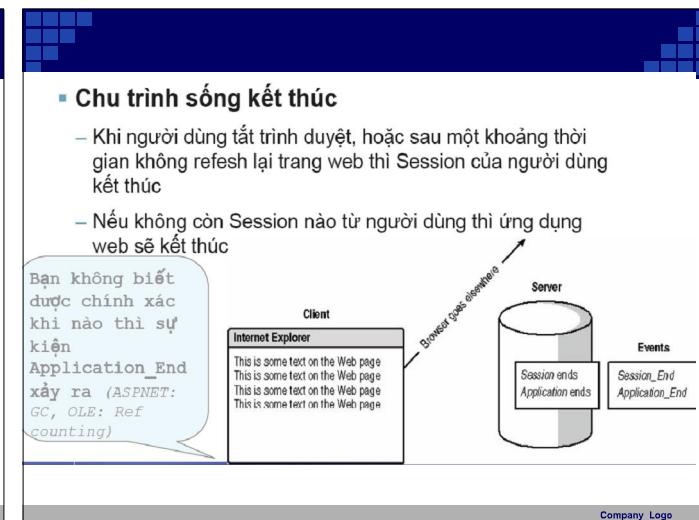
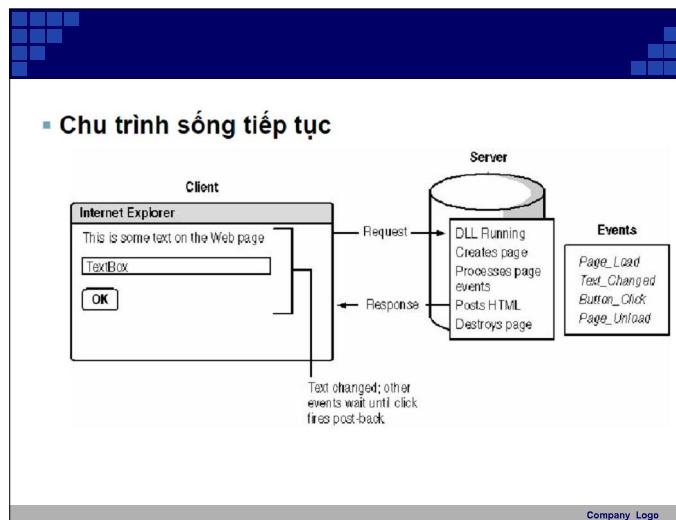
- Bắt đầu khi một trình duyệt yêu cầu 1 trang web từ ứng dụng, gọi là Session
- Ứng dụng web vẫn chạy nếu như nó vẫn còn Session đang hoạt động
- Chu trình sống của 1 Web Form chỉ tồn tại trong 1 khoảng thời gian ngắn

Company Logo

**Chu trình sóng tiếp tục**

- Người dùng tương tác với giao diện web (gõ vào text box, đánh dấu chọn các check box...) cho đến khi kích hoạt một sự kiện **post-back** (nhấn button ...)
- Dữ liệu của trang (**view state**) được gửi về cho server
- Khi server nhận được view state
  - Nó tạo ra thẻ hiện mới của Web Form
  - Điền dữ liệu vào view state
  - Xử lý các sự kiện xảy ra
  - Trả kết quả HTML về cho trình duyệt và **hủy** thẻ hiện của Web Form

Company Logo



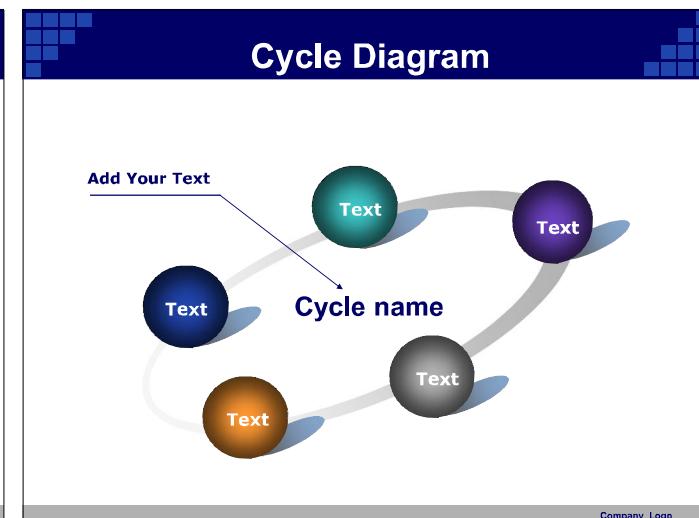
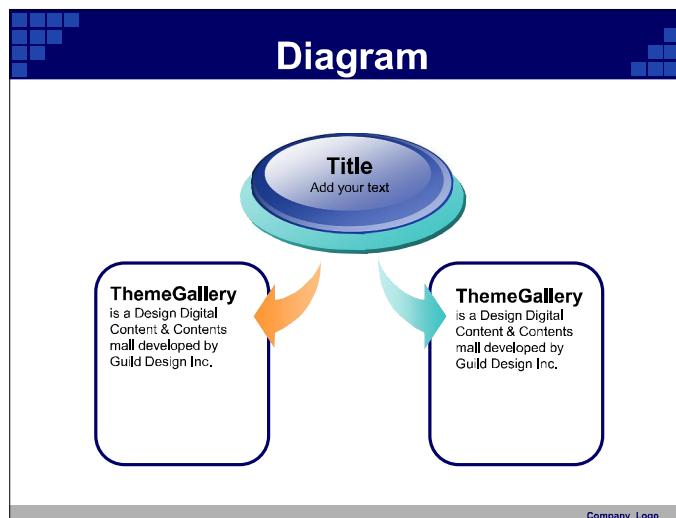
Company Logo

## Hot Tip

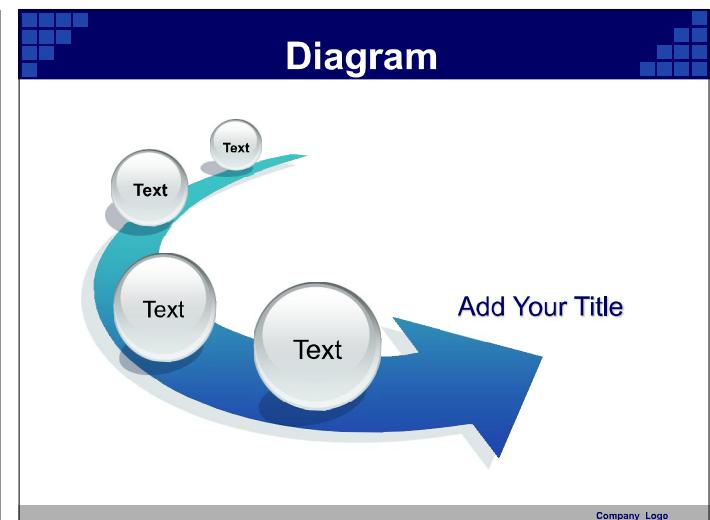
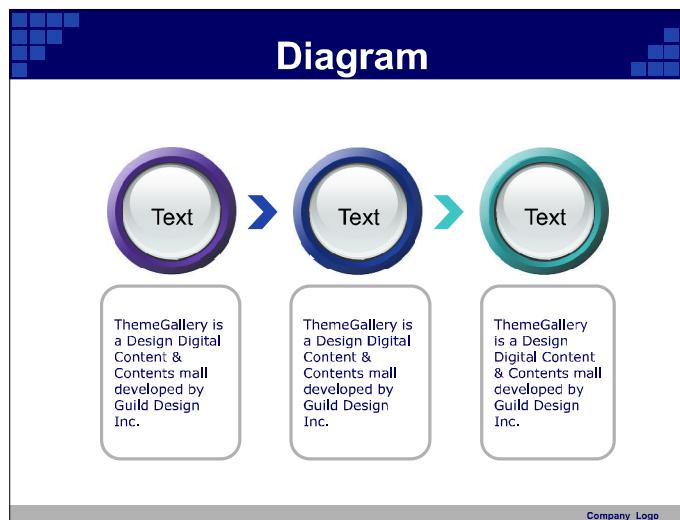
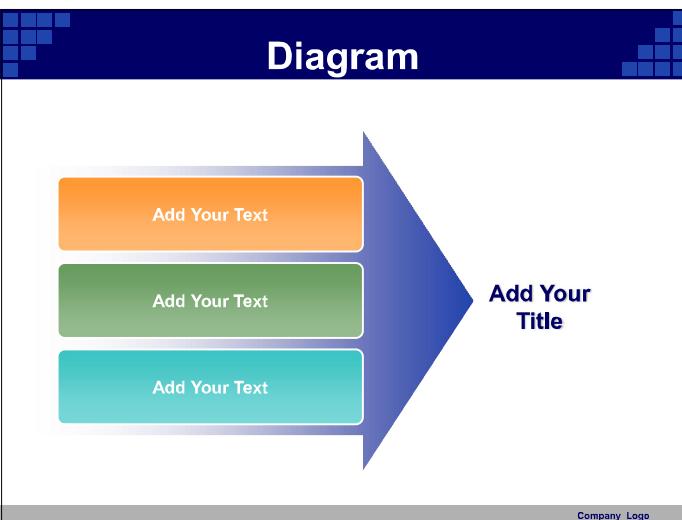
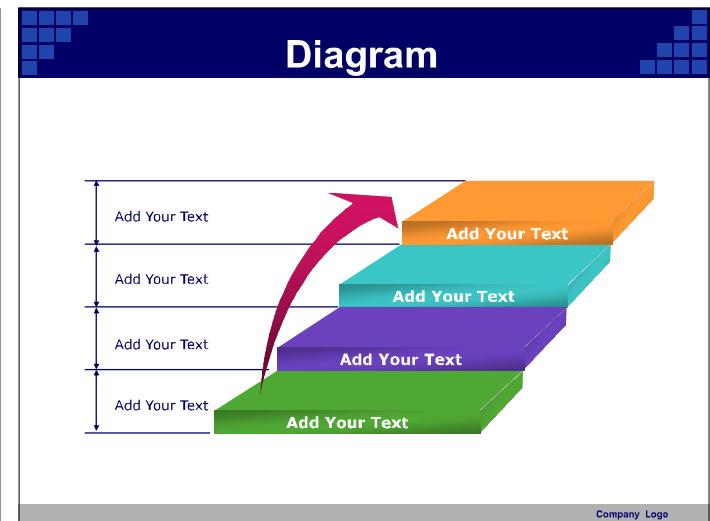
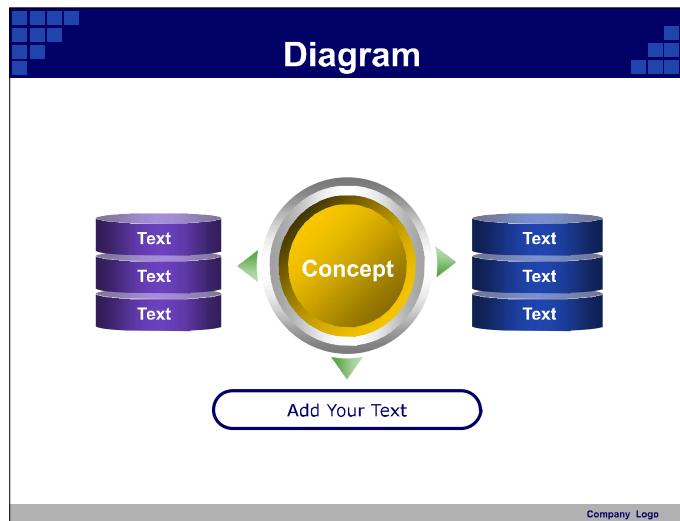
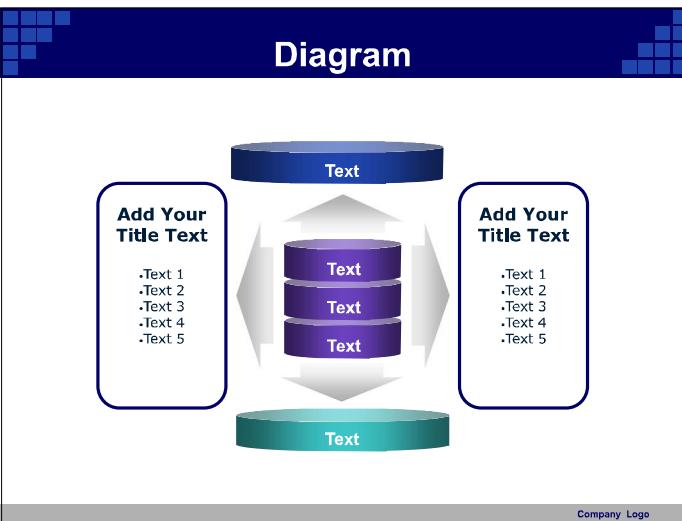
❖ How do I incorporate my logo to a slide that will apply to all the other slides?

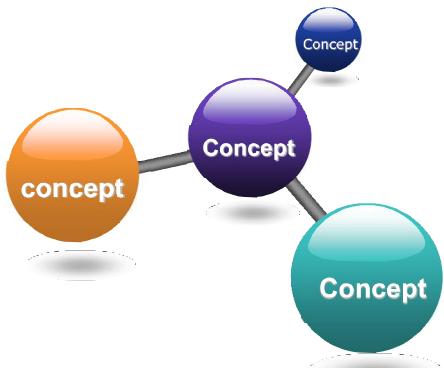
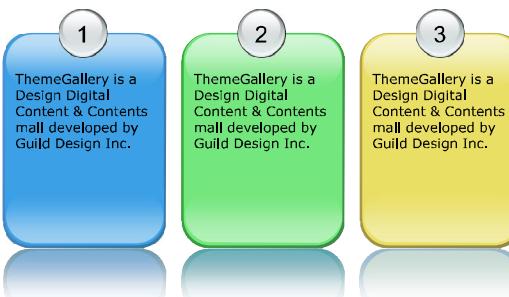
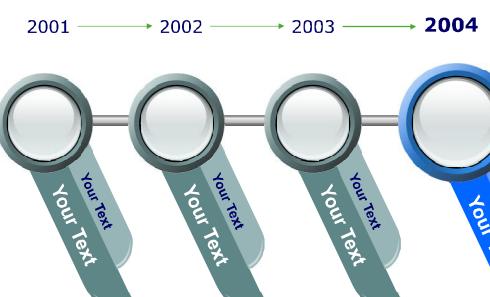
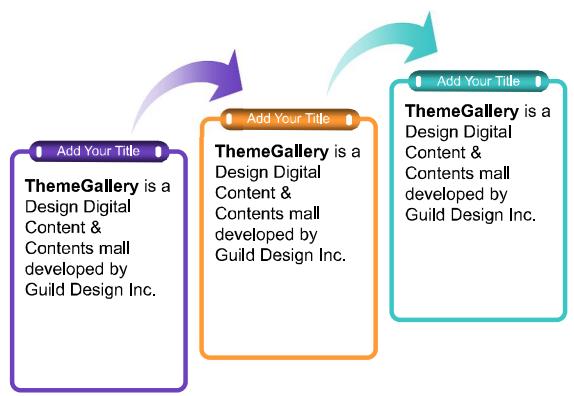
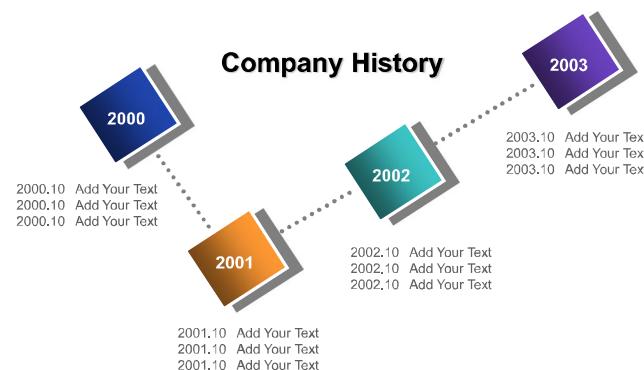
- On the [View] menu, point to [Master], and then click [Slide Master] or [Notes Master]. Change images to the one you like, then it will apply to all the other slides.

Company Logo



Company Logo



**Marketing Diagram****Diagram****Diagram****Diagram****Diagram****Table**

Title	Title	Title	Title	Title
O	O	O	O	O
O	O	O	O	O
O	O	O	O	O
O	O	O	O	O
O	X	O	X	O

