

## Chủ đề 3. VIEWS

---

- Cho phép các thẻ HTML hiển thị trên ứng dụng
- Cung cấp nhiều helper giúp dễ dàng tạo các thẻ HTML
- Phổ biến nhất là mã Razor luôn bắt đầu bằng @
- ViewBag/ViewData và Model được sử dụng để chia sẻ dữ liệu giữa Controller và View

# Controller truyền dữ liệu đến View:

---

## - **ViewBag** (dynamic type):

- Action: `ViewBag.Message = "Hello World!";`
- View: `@ViewBag.Message`

## - **Return views:**

- Action: `return View(model);`
- View: `@model ModelDataType;`

## - **ViewData**

- `ViewData["message"] = "Hello World!";`
- View: `@ViewData["message"]`

## Ví dụ: ViewBag, ViewData

---

```
public ActionResult Detail()
{
    ViewBag.Id = "SV001";
    ViewBag.Name = "Nguyễn Anh Tuấn";
    ViewData["Marks"] = 9.5;
    return View();
}
```

**ViewBag.Id ~  
ViewData["Id"]**

```
<h2>student Detail</h2>
<ul>
    <li>Id: @ViewBag.Id</li>
    <li>Name: @ViewData["Name"]</li>
    <li>Marks: @ViewBag.Marks</li>
</ul>
```

# Ví dụ: View Model

```
public ActionResult Detail()
{
    // Tạo đối tượng
    var model = new StudentInfo
    {
        Id = "SV001",
        Name = "Nguyễn Anh Tuấn",
        Marks = 9.5
    };
    // Truyền đối tượng model cho view
    return View(model);
}
```

Action

```
@model Mvc5.Models.StudentInfo



## Student Detail



- Id: @Model.Id
- Name: @Model.Name
- Marks: @Model.Marks

```

View

Model

```
public class StudentInfo
{
    public string Id { get; set; }
    public string Name { get; set; }
    public double Marks { get; set; }
}
```

Student Detail

- Id: SV001
- Name: Nguyễn Anh Tuấn
- Marks: 9.5

## Razor:

---

- Razor là ngôn ngữ ngắn gọn, rõ ràng và hữu ích cho phép tạo ra các ứng dụng ASP.NET MVC.
- Sử dụng **@ { lệnh }** để viết một mã lệnh thực thi phía server
- Được viết chung với mã HTML phía server
- Sử dụng từ khóa **var** để khai báo biến
- Sử dụng **;** để kết thúc một câu lệnh
- Khối lệnh **@ { khối lệnh }**

## Trong khối lệnh @ {...} là mã C# hoặc VB.NET trộn lẫn HTML

---

```
<!-- Khối lệnh đơn -->
@{ var message = "Hello World"; }

<!-- Biểu thức nội tuyến -->
<p>Giá trị của message là: @message</p>

<!-- Khối nhiều dòng mã lệnh -->
#{@
    var greeting = "Welcome to our site!";
    var weekDay = DateTime.Now.DayOfWeek;
    var greetingMessage = greeting + " Today is: " + weekDay;
}
<p>Lời chào là: @greetingMessage</p>
```

## Vòng lặp

---

- Razor sử dụng cấu trúc vòng lặp giống hoàn toàn của ngôn ngữ C#

- while
- for
- do...while
- foreach

# Cú pháp:

---

Khối mã	<pre>@{     int x = 123;     string y = "because.&gt;"; }</pre>
Biểu thức (đã mã hóa HTML)	<pre>&lt;span&gt;@model.Message&lt;/span&gt;</pre>
Biểu thức (chưa mã hóa HTML)	<pre>&lt;span&gt;@Html.Raw(model.Message)&lt;/span&gt;</pre>
Kết hợp text và HTML	<pre>@foreach (var item in items) {     &lt;span&gt;@item.Prop&lt;/span&gt; }</pre>
Trộn code và text	<pre>@if (foo) {     &lt;text&gt;Plain Text&lt;/text&gt; }</pre>
Trộn code và text	<pre>@if (foo) {     @:Plain Text is @bar }</pre>

Khối using	<pre>@ using (Html.BeginForm()) {     &lt;input type="text" value="input here"&gt; }</pre>
Địa chỉ email	Hi philha@example.com
Biểu thức (tường minh)	<pre>&lt;span&gt;ISBN@(isbnNumber)&lt;/span&gt;</pre>
Mã hóa ký hiệu @	<pre>&lt;span&gt;In Razor, you use the @@foo to display the value of foo&lt;/span&gt;</pre>
Chú thích phía server	<pre>@* This is a server side multiline comment *@</pre>
Trộn biểu thức và text	Hello @title. @name.

## Helper:

---

- ❑ Helper là các thành phần sinh giao diện web phù hợp buộc dữ liệu với model để duy trì thông tin trên các thành phần đó.
- ❑ Đơn giản việc viết mã sinh giao diện
- ❑ Helper được chia làm 1 số nhóm
  - Liên kết
  - Form
  - Sinh giao diện từ model
  - Kiểm lỗi

**@Html.TextBox()**  
**@Html.ActionLink()**  
**@Html.Format()**

## HyperLink Helper:

---

- ❑ `@Html.ActionLink()` được sử dụng để **sinh liên kết**

```
@Html.ActionLink("Giới thiệu", "About")
<a href="/Home/About">Giới thiệu</a>
```

- ❑ `@Html.ActionLink()` nhận một số tham số:

- **linkText** – nhãn của liên kết
  - **actionName** – tên action
  - **routeValues** – tập các giá trị truyền đến action.
  - **controllerName** – tên controller
  - **htmlAttributes** – tập thuộc tính HTML của thẻ `<a>`

- ❑ Ví dụ:

```
@Html.ActionLink("Edit Record", "Edit", new { Id=3})
<a href="/Store/Edit/3">Edit Record</a>
```

- ❑ Liên kết chứa ảnh

```
<a href="@Url.Action("Delete")">
</a>
```

## Form Field Helper:

---

Helper	HTML
<code>@Html.BeginForm()</code>	Sinh thẻ <form> bắt đầu
<code>@Html.EndForm()</code>	Sinh thẻ </form> kết thúc
<code>@Html.CheckBox()</code>	Sinh thẻ <input type="checkbox">
<code>@Html.Hidden()</code>	Sinh thẻ <input type="hidden">
<code>@Html.Password()</code>	Sinh thẻ <input type="password">
<code>@Html.RadioButton()</code>	Sinh thẻ <input type="radio">
<code>@Html.TextArea()</code>	Sinh thẻ <textarea></textarea>
<code>@Html.TextBox()</code>	Sinh thẻ <input type="text">
<code>@Html.DropDownList()</code>	Sinh thẻ <select><option></select>
<code>@Html.ListBox()</code>	Sinh thẻ <select multiple><option></select>

## Ví dụ:

Full Name

```
@{Html.BeginForm("Action", "Controller");}
    <div>Full Name</div>
    @Html.TextBox("FullName")
```

Password

```
<div>Password</div>
@Html.Password("Password")
```

Photo

Không có tệp

```
<div>Photo</div>
<input name="Photo" type="file" />
```

Married Status

**Single**

```
<div>Married Status</div>
<label>@Html.CheckBox("Status") Single</label>
```

Gender

**Male**  **Female**

```
<div>Gender</div>
<label>@Html.RadioButton("Gender", true) Male</label>
<label>@Html.RadioButton("Gender", false) Female</label>
```

Description

```
<div>Description</div>
@Html.TextArea("Description")
```

```
@Html.Hidden("Active")
```

```
<hr />
```

Submit

```
<input type="submit" value="Submit" />
@{Html.EndForm();}
```

# DropDownList và ListBox:

---

```
List<Mail> Mails = new List<Mail>{
    new Mail {
        To = "sender1@gmail.com",
        Subject = "I love you"
    },
    new Mail {
        To = "sender2@gmail.com",
        Subject = "I miss you"
    }
};
ViewBag.Mails = new SelectList(Mails, "To", "Subject");
```

```
@using (Html.BeginForm()){
    @Html.Label("Mails", "E-Mail:");
    @Html.DropDownList("Mails", "Select an email")
    <hr />
    @Html.Label("Mails", "E-Mail:");
    @Html.ListBox("Mails")
}
```

# Sinh mã HTML:

---

**Html.ListBox**

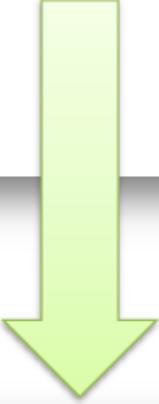
**Html.DropDownList**

```
<form action="/" method="post">
    <label for="Mails">E-Mail:</label>
    <select id="Mails" multiple="multiple" name="Mails">
        <option value="sender1@gmail.com">I love you</option>
        <option value="sender2@gmail.com">I miss you</option>
    </select>
    <hr />

    <label for="Mails">E-Mail:</label>
    <select id="Mails" name="Mails">
        <option value="">Select an email</option>
        <option value="sender1@gmail.com">I love you</option>
        <option value="sender2@gmail.com">I miss you</option>
    </select>
</form>
```

## Sinh <Form>

---

```
@using (Html.BeginForm("Register", "Member")) {  
    ... nội dung form ...  
}  
  
<form action="/Member/Register" method="post">  
    ... nội dung ...  
</form>
```

# Helper định dạng:

---

Helper	Mô tả
@Html.FormatValue (value, format)	Định dạng một giá trị số, chuỗi hoặc thời gian
@String.Format(format, value1, value2...)	Định dạng nhiều giá trị hỗ hợp
@Html.Raw (html)	Giải mã chuỗi đã mã hóa HTML

- Số bình thường: 12345.8765
- Phân nhóm: 12,345.877
- Tiền tệ: \$12,345.88
- Phản trǎm: 72.00 %

- Ngày bình thường: 5/27/2014 9:26:09 PM
- Định dạng D: Tuesday, May 27, 2014
- Định dạng ISO: 2014-05-27
- Định dạng English: 05/27/2014
- Định dạng 24 giờ: 21:26:09
- Định dạng 12 giờ: 09:26:09 PM

- Có mã hóa HTML: <strong>Hello</strong>
- Không mã hóa HTML : Hello

# Định dạng số:

Ký hiệu	Mô tả
{0:C}	Currency – tiền tệ theo ngôn ngữ
{0:P}	Percent – số phần trăm
{0:#,###.##0}	Number – số phân nhóm và 3 số lẻ

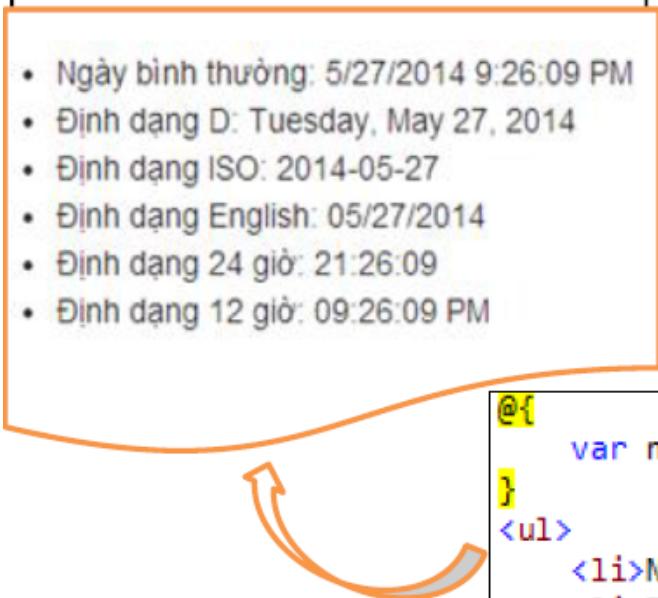
```
@{
    var number1 = 12345.8765;
    var number2 = 0.72;
}
<ul>
    <li>Số bình thường: @number1</li>
    <li>Phân nhóm: @Html.FormatValue(number1, "{0:#,###.##0}")</li>
    <li>Tiền tệ: @Html.FormatValue(number1, "{0:c}")</li>
    <li>Phần trăm: @Html.FormatValue(number2, "{0:p}")</li>
</ul>
```



- Số bình thường: 12345.8765
- Phân nhóm: 12,345.877
- Tiền tệ: \$12,345.88
- Phần trăm: 72.00 %

# Định dạng thời gian:

Ký hiệu	Mô tả
{0:D}	Date – theo ngôn ngữ được chọn
{0:MMMM-dd-yyyy hh:mm:ss tt}	<ul style="list-style-type: none"><li>• Ngày bình thường: 5/27/2014 9:26:09 PM</li><li>• Định dạng D: Tuesday, May 27, 2014</li><li>• Định dạng ISO: 2014-05-27</li><li>• Định dạng English: 05/27/2014</li><li>• Định dạng 24 giờ: 21:26:09</li><li>• Định dạng 12 giờ: 09:26:09 PM</li></ul> <ul style="list-style-type: none"><li>✓ M,MM,MMM,MMMM: tháng 1, 2 ký tự số, 3 ký tự viết tắt, tên tháng đầy đủ</li><li>✓ d,dd: ngày 1, 2 ký tự</li><li>✓ yy,yyyy: năm 2, 4 ký tự số</li><li>✓ H, HH, h, hh: 1,2 ký tự giờ 24 hoặc 12 giờ mỗi ngày</li><li>✓ m,mm: 1,2 ký tự số phút</li><li>✓ s,ss: 1,2 ký tự số giây</li><li>✓ tt: 2 ký tự sáng/chiều</li></ul>



```
@{  
    var now = DateTime.Now;  
}  
  
<ul>  
    <li>Ngày bình thường: @now</li>  
    <li>Định dạng D: @Html.FormatValue(now, "{0:D}")</li>  
    <li>Định dạng ISO: @Html.FormatValue(now, "{0:yyyy-MM-dd}")</li>  
    <li>Định dạng English: @Html.FormatValue(now, "{0:MM/dd/yyyy}")</li>  
    <li>Định dạng 24 giờ: @Html.FormatValue(now, "{0:HH:mm:ss}")</li>  
    <li>Định dạng 12 giờ: @Html.FormatValue(now, "{0:hh:mm:ss tt}")</li>  
</ul>
```

## Mã hóa HTML:

---

- Có mã hóa HTML: <strong>Hello</strong>
- Không mã hóa HTML : Hello

```
@{
    var chuoi = "<strong>Hello</strong>";
}
<ul>
    <li>Có mã hóa HTML: @chuoi</li>
    <li>Không mã hóa HTML : @Html.Raw(chuoi)</li>
</ul>
```

## Sinh giao diện theo Model:

- ❑ Dựa vào các đặc điểm của thuộc tính trong lớp model để sinh ra giao diện người dùng.
  - Sinh các control tương minh
  - Sinh các control ngầm định

```
public class Student
{
    [DisplayName("Mã sinh viên")]
    public String Id { get; set; }
    [DisplayName("Mật khẩu")]
    public String Password { get; set; }
    [DisplayName("Họ và tên")]
    public String FullName { get; set; }
    [DisplayName("Giới tính")]
    public bool Gender { get; set; }
    [DisplayName("Ngày sinh")]
    public DateTime Birthday { get; set; }
    [DisplayName("Ghi chú")]
    public String Notes { get; set; }
}
```

The diagram illustrates the process of generating a user interface from a C# class. On the left, a code box contains the `Student` class definition with various properties and their `[DisplayName]` attributes. An orange arrow points from this code to a right-hand box labeled "Đăng ký thành viên". This registration form contains fields for "Mã sinh viên", "Mật khẩu", "Họ và tên" (with the value "Nguyễn Nghiêm" entered), "Giới tính" (with radio buttons for "Nam" and "Nữ" visible), "Ngày sinh" (with the value "9/1/1971 12:00:00 AM" entered), and "Ghi chú". A "Register" button is at the bottom.

Đăng ký thành viên

Mã sinh viên	<input type="text"/>
Mật khẩu	<input type="password"/>
Họ và tên	Nguyễn Nghiêm
Giới tính	<input checked="" type="radio"/> Nam <input type="radio"/> Nữ
Ngày sinh	9/1/1971 12:00:00 AM
Ghi chú	<input type="text"/>

Register

## Sinh UI tự động:

- ☐ Chỉ định loại control đối với các thuộc tính

Helper	Mô tả
@Html.TextBoxFor (m => m.Id)	<input type="text" name="Id" id="Id">
@Html.PasswordFor (m => m.Pwd)	<input type="password" name="Pwd" id="Pwd">
@Html.TextAreaFor (m => m.Notes)	<textarea name="Notes" id="Notes"></textarea>
@Html.CheckBoxFor (m => m.Status)	<input type="checkbox" name="Status" id="Status">
@Html.RadioButtonFor (m => m.Gender)	<input type="radio" name="Gender" id="Gender">
@Html.HiddenFor (m=> m.Name)	<input type="hidden" name="Name" id="Name">
@Html.DropDownListFor (m=> m.Blood)	<select id="Blood" name="Blood">...</select>
@Html.ListBoxFor (m=> m.Jobs)	<select id="Jobs" name="Jobs" multiple>...</select>
@Html.LabelFor (m=> m.Name)	<label for="Name"> Name </label>

```
@model Mvc5CodeDemo.Models.Student
<h2>Đăng ký thành viên</h2>
@using (Html.BeginForm())
{
    <table><tr>
        <td>@Html.LabelFor(m => m.Id)</td>
        <td>@Html.TextBoxFor(m => m.Id)</td>
    </tr><tr>
        <td>@Html.LabelFor(m => m.Password)</td>
        <td>@Html.PasswordFor(m => m.Password)</td>
    </tr><tr>
        <td>@Html.LabelFor(m => m.FullName)</td>
        <td>@Html.TextBoxFor(m => m.FullName)</td>
    </tr><tr>
        <td>@Html.LabelFor(m => m.Gender)</td>
        <td>
            <label>@Html.RadioButtonFor(m => m.Gender, true) Nam</label>
            <label>@Html.RadioButtonFor(m => m.Gender, false) Nữ</label>
        </td>
    </tr><tr>
        <td>@Html.LabelFor(m => m.Birthday)</td>
        <td>@Html.TextBoxFor(m => m.Birthday)</td>
    </tr><tr>
        <td>@Html.LabelFor(m => m.Notes)</td>
        <td>@Html.TextAreaFor(m => m.Notes)</td>
    </tr><tr>
        <td>&nbsp;</td>
        <td><input type="submit" value="Register" /></td>
    </tr></table>
}
```

Kiểu của Model

Sinh <label for="Id">Mã  
sinh viên</label>

Sinh <input type="text" name="Id"  
id="Id"> từ thuộc tính Id của Model

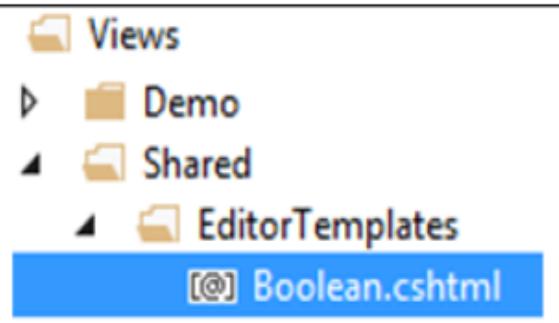
## Sinh UI ngầm định:

- Tự sinh loại control phù hợp với đặc điểm của thuộc tính của lớp model.

Helper	Mô tả
Html.EditorFor(m=>m.Property)	Sinh 1 control cho 1 thuộc tính.
Html.EditorForModel()	Sinh toàn form theo các thuộc tính của Model
Html.Editor(object)	Sinh toàn form theo các thuộc tính của Object đặt trong ViewBag

Bổ sung thêm template để hiển thị giới tính dạng RadioButtonList bằng cách thêm Boolean.cshtml vào thư mục Views/Shared/EditorTemplates

```
@model Boolean  
<label>@Html.RadioButton("Gender", true, @Model == true)  
Nam</label>  
<label>@Html.RadioButton("Gender", false, @Model == false)  
Nữ</label>
```



```
public class Student
{
    [DisplayName("Mã sinh viên")]
    public String Id { get; set; }
    [DisplayName("Mật khẩu"), DataType(DataType.Password)]
    public String Password { get; set; }
    [DisplayName("Họ và tên")]
    public String FullName { get; set; }
    [DisplayName("Giới tính")]
    public bool Gender { get; set; }
    [DisplayName("Ngày sinh")]
    public DateTime Birthday { get; set; }
    [DisplayName("Ghi chú"), DataType(DataType.MultilineText)]
    public String Notes { get; set; }
}
```

```
@model Mvc5CodeDemo.Models.Student
@using (Html.BeginForm())
{
    @Html.EditorForModel()
    <input type="submit" value="Register" />
}
```

The diagram illustrates the relationship between a C# model class, an MVC view, and the resulting rendered form.

**C# Model Class:** The code defines a `Student` class with properties: `Id`, `Password`, `FullName`, `Gender`, `Birthday`, and `Notes`. Each property is annotated with `[DisplayName]` attributes and specific `DataType` annotations like `(DataType.Password)` for the password and `(DataType.MultilineText)` for notes.

**MVC View:** The view uses `@Html.EditorForModel()` to generate form fields for all properties defined in the `Student` model. It also includes a `<input type="submit" value="Register" />` button.

**Rendered Form:** The final output is a registration form with the following fields:

- Mã sinh viên (Input field)
- Mật khẩu (Input field)
- Họ và tên (Input field, containing "Nguyễn Nghiêm")
- Giới tính (Radio buttons: Nam (selected) and Nữ)
- Ngày sinh (Input field, containing "9/1/1971 12:00:00 AM")
- Ghi chú (Text area)
- Register (Submit button)

# Kiểu Control:

---

<b>DataType</b>	<b>Mô tả</b>
DataType.CreditCard	Cần cho phép nhập số thẻ tín dụng
DataType.Currency	Hiển thị và tiếp nhận dạng tiền tệ theo địa phương được chọn
DataType.Date	Hiển thị và tiếp nhận dạng ngày theo địa phương được chọn
DataType.DateTime	Hiển thị và tiếp nhận dạng ngày và giờ theo địa phương được chọn
DataType.Duration	Sinh slider trên thiết bị di động
DataType.EmailAddress	Cần cho phép nhập email
DataType.Html	Cho phép nhập mã html
DataType.ImageUrl	Cần cho phép nhập địa chỉ ảnh
DataType.MultilineText	Sinh <textarea>
DataType.Password	Sinh <input type="password">
DataType.PhoneNumber	Sinh phần tử nhập số điện thoại trên thiết bị di động
DataType.PostalCode	Cần cấp nhận dạng postal code
DataType.Text	Nhập văn bản thông thường
DataType.Time	Ô nhập thời gian
DataType.Upload	Ô nhập upload file
DataType.Url	Cần nhấp nhận địa chỉ tài nguyên

## Hiện thị thông tin:

---

- `@Html.DisplayNameFor (m=> m.Property)`
  - Hiển thị tên của thuộc tính Property
- `@Html.DisplayFor (m=>m.Property)`
  - Hiển thị giá trị cho thuộc tính Property
- `@Html.DisplayForModel ()`
  - Hiển thị giá trị của tất cả các thuộc tính
- `@Html.Display ("Mail")`
  - Hiển thị giá trị của tất cả các thuộc tính của đối tượng trong ViewData hay ViewBag

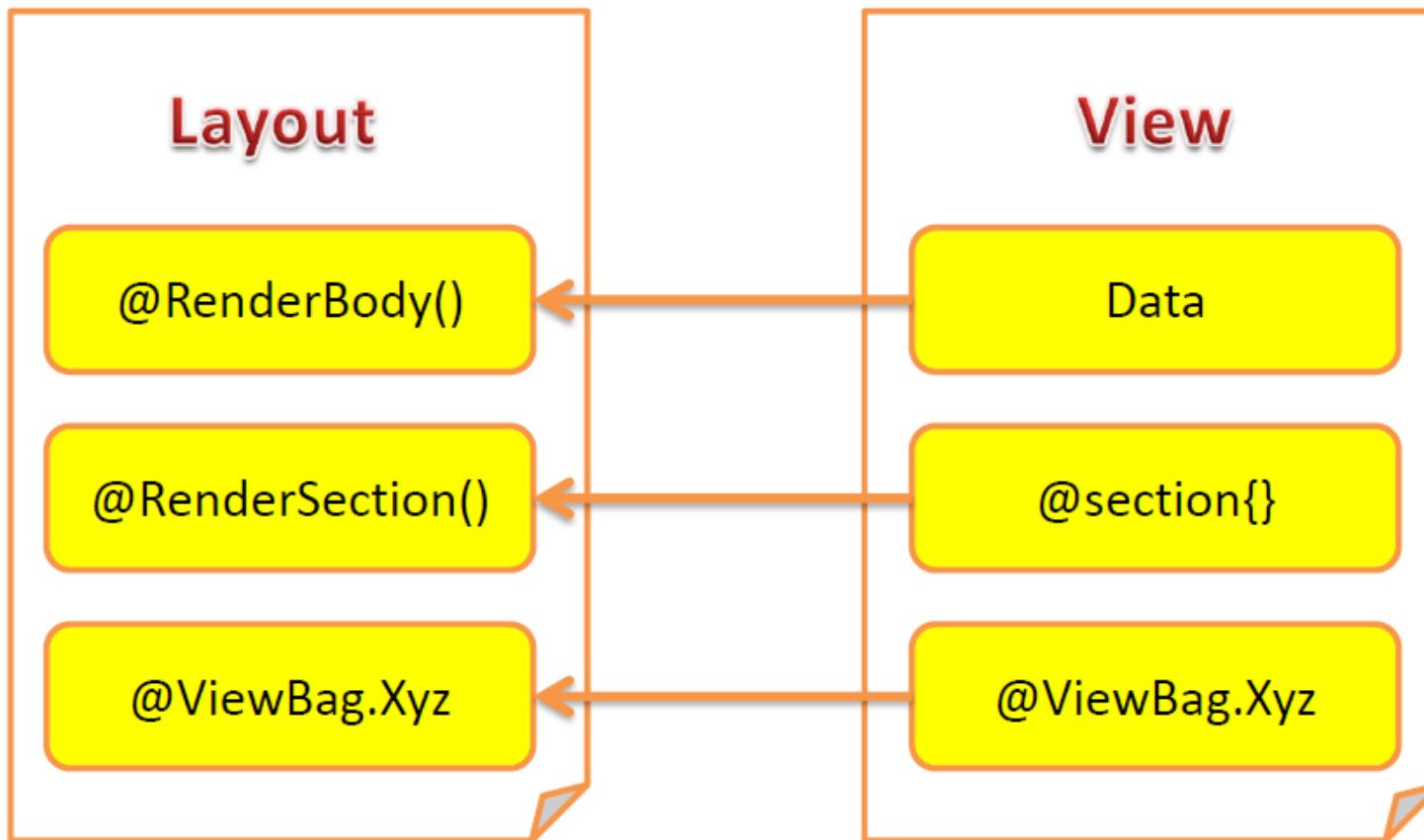
## Layout

---

- ❑ Layout trong MVC là Masterpage trong Webform
- ❑ Layout được thiết kế và sử dụng các style CSS đã học
- ❑ Layout trong MVC chứa trong thư lục **Views/Shared** với phần mở rộng là **.cshtml**

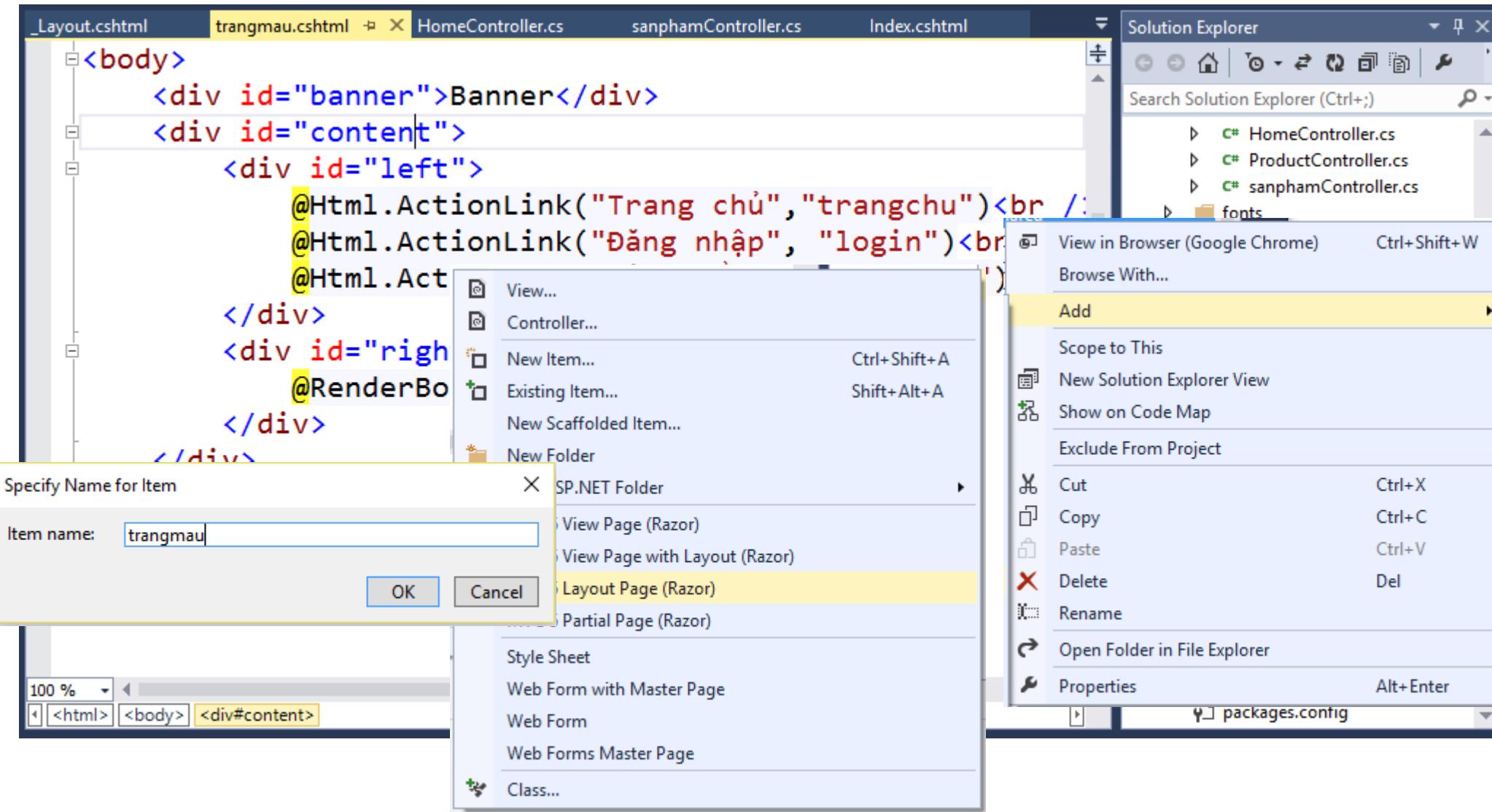
## Layout chứa

- ☞ **Một và chỉ một @RenderBody()** để giữ chỗ cho nội dung trong view
- ☞ **Không hoặc nhiều @RenderSection()** để giữ chỗ cho các phần được đánh dấu @section trong view



# Cách tạo Layout

- Right Click View\Shared → Chọn Add → Chọn MVC Layout Page(Razor)



# Cách tạo Layout

---

- Layout được tạo như sau:

@RenserBody là phần nội dung được thay đổi

```
<!DOCTYPE html>

<html>
  <head>
    <meta name="viewport" content="width=device-width" />
    <title>@ViewBag.Title</title>
  </head>
  <body>
    <div>
      @RenderBody()
    </div>
  </body>
</html>
```

# Thiết lập layout cơ bản

---

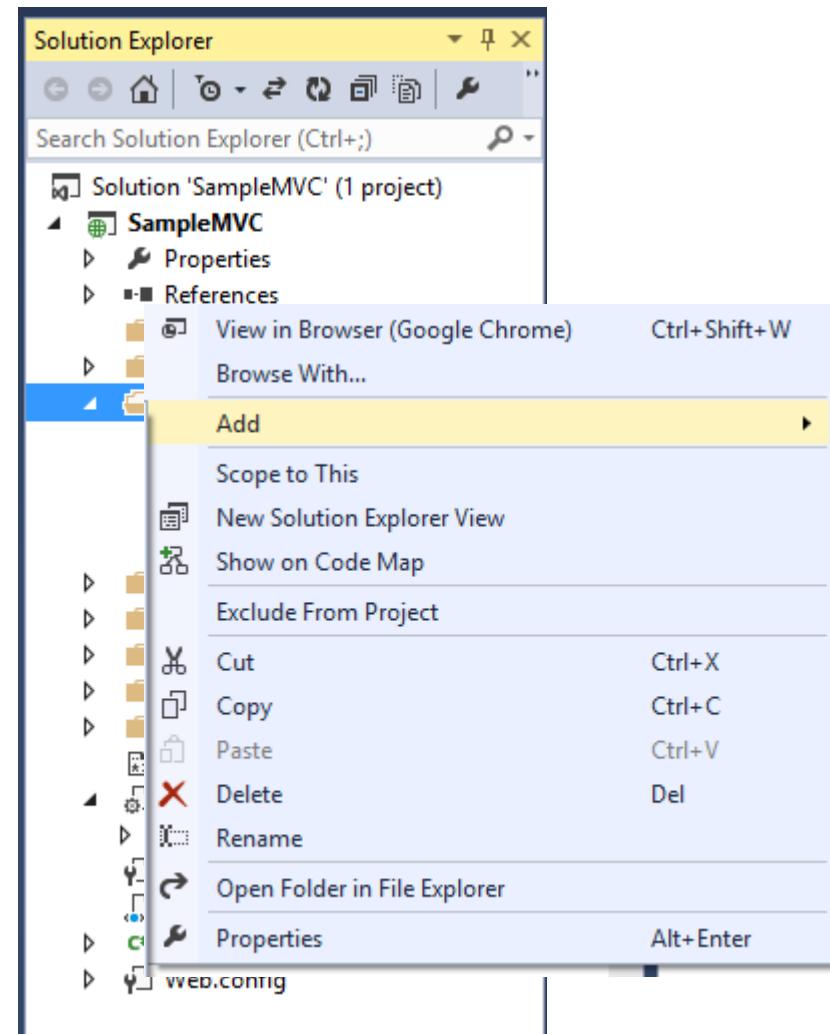
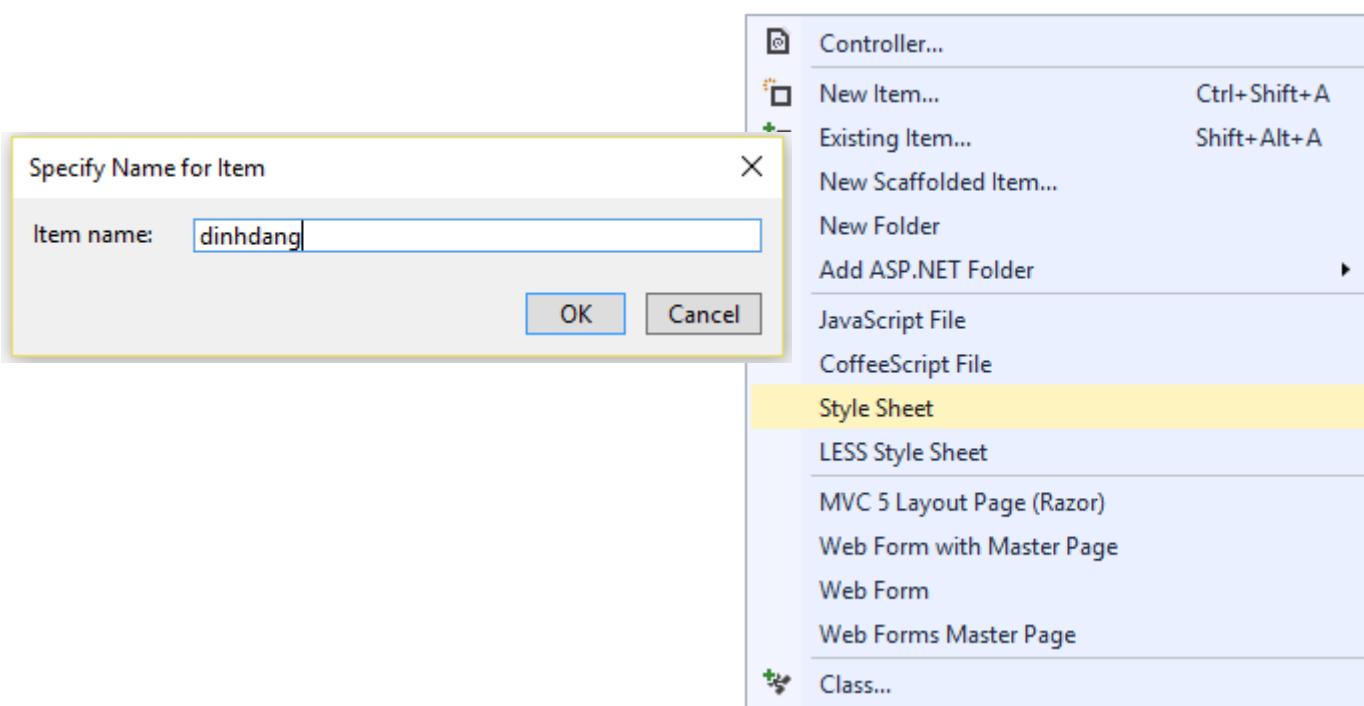


```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>@ViewBag.Title</title>

    <link href="~/Content/dinhdang.css" rel="stylesheet" />
</head>
<body>
    <div id="banner">Banner</div>
    <div id="content">
        <div id="left">
            @Html.ActionLink("Trang chủ", "trangchu")<br />
            @Html.ActionLink("Đăng nhập", "login")<br />
            @Html.ActionLink("Sản phẩm", "Chitietsp")
        </div>
        <div id="right">
            @RenderBody()
        </div>
    </div>
    <div id="footer">Copy right ...</div>
</body>
</html>
```

# Áp dụng CSS cho layout

- Trong thư mục Content → Right Click → Add Style Sheet
- Đặt tên cho file css



```
body {  
    margin-top:0px;  
    margin-left:100px;  
}  
  
#banner {  
    width:800px;  
    background-color: aqua;  
    height:80px;  
    border:1px solid black;  
    float:left;  
}  
  
#footer {  
    width:800px;  
    background-color: violet;  
    height:50px;  
    border:1px solid black;  
    float:left;  
}  
  
#content {  
    width:800px;  
    background-color: pink;  
    height:400px;  
    float:left;  
}  
  
#left {  
    width:150px;  
    background-color: aquamarine;  
    border:1px solid black;  
    height:398px;  
    float:left;  
}  
  
#right {  
    width:645px;  
    background-color:beige;  
    border:1px solid black;  
    height:398px;  
    float:left;  
}  
  
#container {  
    width:800px;  
    height:auto;  
    float:left;  
    margin:auto;  
    border:5px solid red;  
}
```

- Nhúng file CSS vào trong layout bằng cách kéo file vào trang layout
- Mã lệnh :

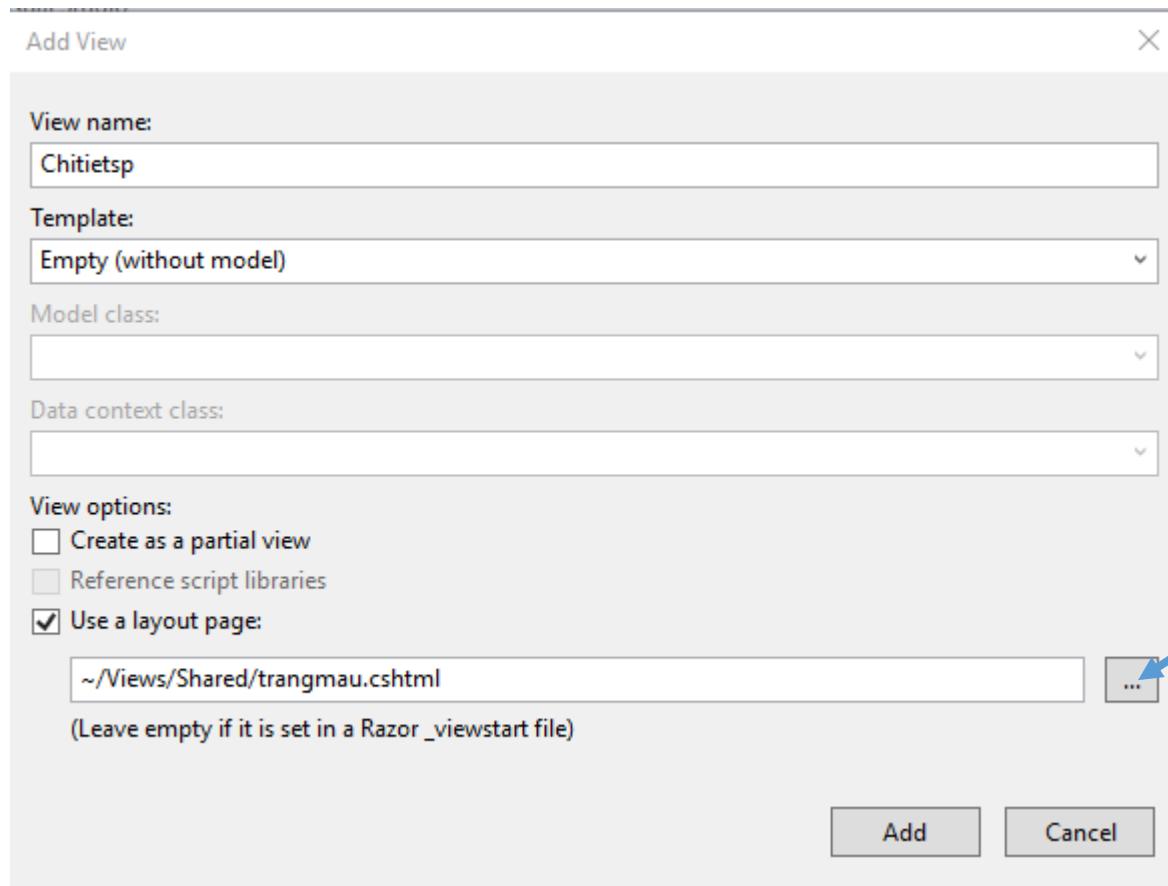
```

dang.css      Site.css      _Layout.cshtml    trangmau.cshtml*  HomeController.cs  sanphamController.cs
  !DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="width=device-width" />
    <title>@ViewBag.Title</title>
    <link href="~/Content/dinhdang.css" rel="stylesheet" />
  </head>
  <body>
    <div id="banner">Banner</div>
    <div id="content">
      <div id="left">
        @Html.ActionLink("Trang chủ", "trangchu")<br />
        @Html.ActionLink("Đăng nhập", "login")<br />
        @Html.ActionLink("Sản phẩm", "Chitietsp")
      </div>
      <div id="right">
        @RenderBody()
      </div>
    </div>
    <div id="footer">Copy right ...</div>
  </body>
</html>

```

# Tạo view với Layout sẵn

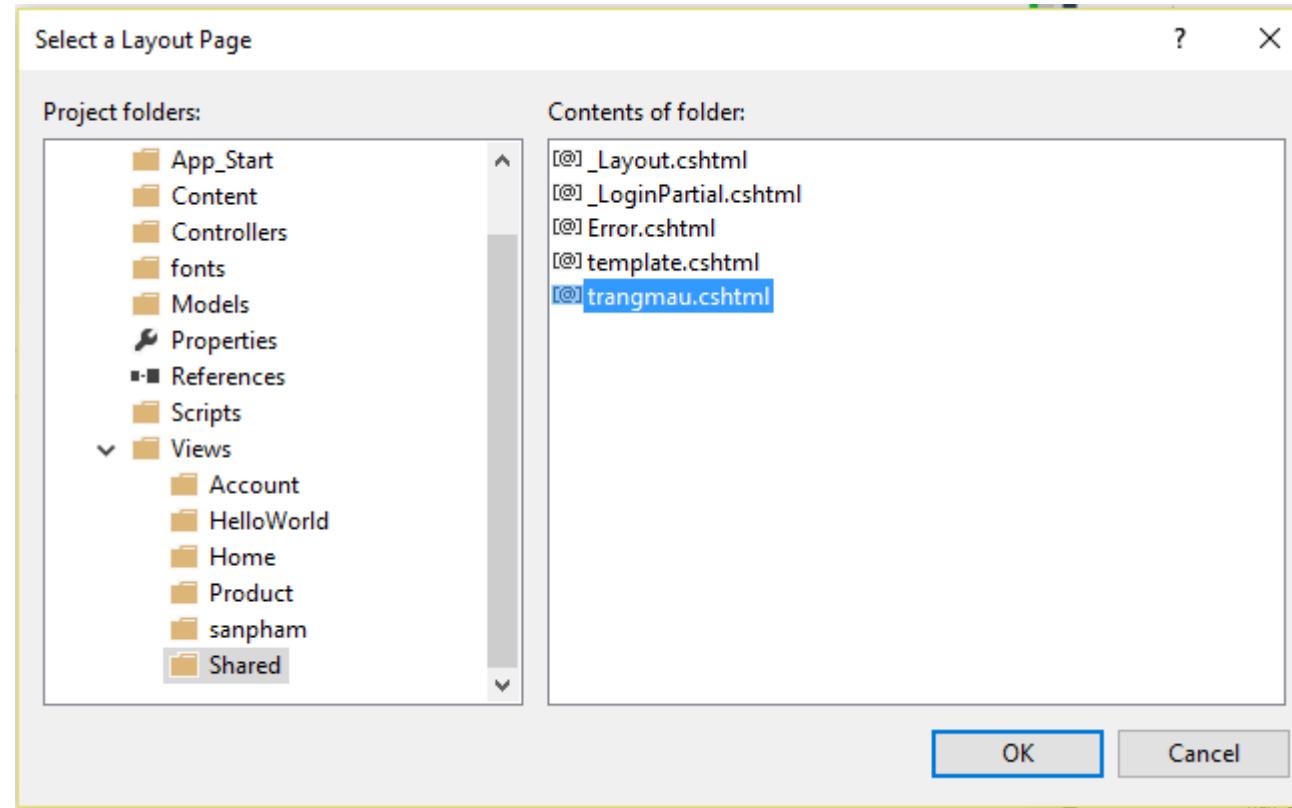
- Trong thư mục View → Add View



Click chọn  
danh sách các  
layout đã tạo

---

## - Chọn trang layout tương ứng



- Tại View có sử dụng layout vừa chọn sẽ có mã lệnh tương ứng

The screenshot shows the Visual Studio IDE interface. On the left is the code editor window titled "Chitietsp.cshtml". The code defines a view model and sets the layout to a shared master page:

```
@{  
    ViewBag.Title = "Chitietsp";  
    Layout = "~/Views/Shared/trangmau.cshtml";}  
  
<h2>Chitietsp</h2>  
■<ul>  
    <li>Mã sản phẩm: @Model.maSP</li>  
    <li>Tên sản phẩm: @Model.tenSP</li>  
    <li>Giá sản phẩm: @Model.giaSP</li>  
</ul>
```

On the right is the "Solution Explorer" window, which lists the project structure. The "Views" folder contains several files, including "Chitietsp.cshtml" which is currently selected.

```
dinhdang.css Site.css trangmau.cshtml* Chitietsp.cshtml _Layout.cshtml trangchu.cshtml  
Solution Explorer  
Search Solution Explorer (Ctrl+;) ▾  
HelloWorldController.cs  
HomeController.cs  
ProductController.cs  
sanphamController.cs  
fonts  
Models  
Scripts  
Views  
Account  
HelloWorld  
Home  
Product  
sanpham  
Chitietsp.cshtml  
DanhsachSP.cshtml  
login.cshtml  
trangchu.cshtml  
Shared
```

# PartialView Helper

---

## ❑ @Html.Action()

- Nhúng một Action
- Action này phải trả về PartialView để *loại bỏ Layout*
- Action này có thể đánh dấu [ChildActionOnly] để *không cho truy xuất trực tiếp*

## ❑ Sử dụng @Html.Partial()

- Nhúng một View *không bao gồm layout*

# Sử dụng @Html.Partial()

## \_Layout.cshtml

❑ @Html.Partial("\_LoginPartial")



```
_LoginPartial.cshtml ✘ X
@if (Request.IsAuthenticated) {
    <text>
        Hello, @Html.ActionLink(User.Identity.Name,
            @using (Html.BeginForm("LogOff", "Account",
                @Html.AntiForgeryToken()))
            <a href="javascript:document.getElementById('logOff').submit()">Log off</a>
    </text>
} else {
    <ul>
        <li>@Html.ActionLink("Register", "Register", "Account")
        <li>@Html.ActionLink("Log in", "Login", "Account")
    </ul>
}
```

# Sử dụng @Html.Action()

The screenshot shows two code editors in Visual Studio. The top editor is 'LayoutController.cs' and the bottom one is '\_Category.cshtml'.

**LayoutController.cs:**

```
[ChildActionOnly]
public ActionResult Category()
{
    return PartialView("_Category", db.Categories);
}
```

**\_Category.cshtml:**

```
@model IEnumerable<Category>




@foreach(var p in Model){
    <li>
        <a href="/Product/Search?CategoryId=@p.Id">
            @p.Name</a>
    </li>
}

```

An orange arrow points from the line of code `@Html.Action("Category", "Layout")` in the LayoutController.cs file to the `@model` declaration in the \_Category.cshtml file.

# BỎ SUNG:

---

- WebGrid
- Button: Text, color, Image
- ActionLink

## WebGrid

---

- Thêm Microsoft.AspNet.WebPages.WebData từ NuGet.

```
@using WebMatrix.Data;
@{
    var db = Database.Open("ModelUD4");//Chuỗi kết nối trong
web.config
    var selectStr = "SELECT * FROM NhanVien";
    var data = db.Query(selectStr);
    var grid = new WebGrid(data);
}
<h1>DANH SÁCH NHÂN VIÊN</h1>
<div id="grid">
    @grid.GetHtml(tableStyle: "table")
</div>
```

## Các thuộc tính thường dùng của WebGrid:

---

- source //*Nguồn dữ liệu*
- rowsPerPage: n //*Phân trang n dòng trên 1 trang*
- defaultSort: “Tên thuộc tính”//*Sắp xếp mặc định*
- canPage: true/false //*Cho phép phân trang?*
- canSort: true/false //*Cho phép sắp xếp?*

## WebGrid GetHtml:

---

```
- tableStyle: "table",
- mode: WebGridPagerModes.Numeric,
- columns: new[] // columns in grid
{
    grid.Column("Tên thuộc tính", "Tên mới hiển thị", định dạng),
    ...
}
```

```
@grid.GetHtml(
    tableStyle: "table",
    mode: WebGridPagerModes.Numeric,
    columns: new[] // columns in grid
{
    grid.Column("MaNV", "Mã nhân viên"), //the model fields to
display
    grid.Column("HoNV", "Họ nhân viên"),
    grid.Column("TenNV", "Tên nhân viên"),
    grid.Column("GioiTinh", "Giới tính", format: (@item) =>
@item.GioiTinh==true?"Nam":"Nữ"),
    grid.Column("NgaySinh", "Ngày sinh", format:@<text>
@item.NgaySinh.ToString("dd/MM/yyyy")</text>),
    grid.Column("DiaChi", "Địa chỉ"),
    grid.Column("MaPB", "Phòng ban"),
    grid.Column("ChucNang", format: (item) =>
Html.ActionLink("Cập nhật", "Edit", new { id = item.MaNV })),
```

```
grid.Column("Chức năng", format: @<text>
    <a href="@Url.Action("Edit",
        new { id = item.MaNV})" class="edit-
btn"><abbr title="Cập nhật dữ liệu"></abbr></a>
    <a href="@Url.Action("Details",
        new { id = item.MaNV})" class="edit-
btn"><abbr title="Xem chi tiết"></abbr></a>
    <a href="@Url.Action("Delete",
        new { id = item.MaNV})" class="edit-btn">
        <abbr title="Xóa dữ liệu">
            </abbr> </a>
    </text>),
}
```

# Kết quả thể hiện trên trình duyệt:

---

## DANH SÁCH NHÂN VIÊN

Mã nhân viên	Họ nhân viên	Tên nhân viên	Giới tính	Ngày sinh	Địa chỉ	Phòng ban	Chức năng	Chức năng
XT	2	3	Nam	01/01/1999	1234	1	Cập nhật	  
2	3	4	Nam	01/01/1999	1234	KD	Cập nhật	  
ZT	4	5	Nữ	01/01/1999	345	KT	Cập nhật	  
6	6	7	Nam	01/01/1999	789	KT	Cập nhật	  

12

## Button:

---



```
<input type="submit" value="Thêm" style="padding-left: 28px; margin-left: 2px; background-image: url('/images/edit.png'); background-repeat: no-repeat; background-position: left; background-size: 20px;" />
```

```
<input type="submit" value="Thêm" style="margin-left: 2px; width: 74px; background-color: blue; color: white" />
```