LẬP TRÌNH WEB

Nguyễn Thị Mai Trang

1

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Nội dung

- Chương 1: Tổng quan
- Chương 2: Ngôn ngữ PHP
- Chương 3: Controller – View – Model
- **Chương 4: HTML Helpers**
- Chương 5: Làm việc với Cơ sở dữ liệu

2

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Chương 4

HTML Helpers

3

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Nội dung

- Giới thiệu HTML Helper
- Form và các control
- Kiểm chứng dữ liệu

4

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.1. Giới thiệu HTML Helpers

- ASP.NET helpers là các thành phần giao diện web có thể truy cập bằng cú pháp Razor.
- Một số thành phần hữu ích trong helper:
  - WebGrid: hiển thị dữ liệu dạng bảng, hỗ trợ định dạng, phân trang, sắp xếp dữ liệu.
  - Chart: hiển thị biểu đồ với nhiều định dạng, nguồn dữ liệu có thể là mảng, bảng CSDL, file
  - WebImage: cung cấp các chức năng để hiển thị hình ảnh trong trang web: lật, xoay, thay đổi kích thước, làm mờ ảnh,...
  - Analytics (Google), FileUpload, Json, LinkShare, Recaptcha
  - Chức năng hỗ trợ tìm kiếm (Bing), mã hóa (Crypto), tạo kết nối với Facebook
  - ...

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Giới thiệu HTML Helpers (tt)

- Lớp HtmlHelper tạo các phần tử html → có thể sử dụng lớp HtmlHelper kết hợp cú pháp Razor thay cho các tag HTML truyền thống.
- Ví dụ:
  - @Html.ActionLink("Create New", "Create")
    thay vì
    Create New.

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Giới thiệu HTML Helpers (tt)

- Một số HtmlHelper thông dụng:

HtmlHelper	Strongly Typed HtmlHelpers	Html Control
Html.ActionLink		Liên kết
Html.TextBox	Html.TextBoxFor	Textbox
Html.TextArea	Html.TextAreaFor	TextArea
Html.CheckBox	Html.CheckBoxFor	Checkbox
Html.RadioButton	Html.RadioButtonFor	Radio button
Html.DropDownList	Html.DropDownListFor	Dropdown, combobox
Html.ListBox	Html.ListBoxFor	multi-select list box
Html.Hidden	Html.HiddenFor	Hidden field
Password	Html.PasswordFor	Password textbox
Html.Display	Html.DisplayFor	Văn bản
Html.Label	Html.LabelFor	Label
Html.Editor	Html.EditorFor	Tạo các điều khiển Html tương ứng với kiểu dữ liệu của thuộc tính trong model

7

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - ActionLink

- **Html. ActionLink()**: tạo phần tử <a>
  - MvcHtmlString ActionLink (HtmlHelper htmlHelper, string linkText, string actionName);

View:

```
<!-- Redirect in Same Controller -->
@Html.ActionLink("Contact Page","Contact")
<br />
<!-- Redirect in Another Controller -->
@Html.ActionLink("Report Page","Index","Report")
```

Chuỗi liên kết

Tên Action

Tên Controller

Mã Html tương ứng:

```
<!-- Redirect in Same Controller -->
<a href="/Home/Contact">Contact Page</a>
<br />
<!-- Redirect in Another Controller -->
<a href="/Report">Report Page</a>
```


8

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - ActionLink

- **Html. ActionLink()**: truyền tham số

```
@Html.ActionLink("Edit link text", "Edit", "StoreManager", new {id=10720}, null)
```

Chuỗi liên kết | Tên Action | Tên Controller | Tham số: id=10720 | Thuộc tính

9

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - RouteLink

- Tương tự Html. ActionLink(), nhưng khác tham số (nhận vào một tên tuyến và không có hai đối số tên Controller và Action), ví dụ:

```
@Html.RouteLink("Link Text", "RouteName"})
```

Chuỗi liên kết | Tên tuyến

10

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Url

- Tương tự ActionLink và RouteLink, nhưng không tạo phần tử <a>, mà trả về URL dưới dạng chuỗi. Có ba Helper: Action, Content và RouteUrl
- **Url.Action**:

Tên Action

Tên Controller

```
View:
<span>
  @Url.Action("Browse", "Store", new { genre = "Jazz" }, null)
</span>
```

Tham số

```
Mã HTML tương ứng:
<span>
  /Store/Browse?genre=Jazz
</span>
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Url

- **Url.RouteUrl**: tương tự Action, nhưng giống như RouteLink, nó nhận vào một tên tuyến và không có hai đối số tên Controller và Action.
- **Url.Content**: cho phép chuyển đổi đường dẫn tương đối của ứng dụng thành đường tuyệt đối. Ví dụ:

```
<script src= "@Url.Content("~/Scripts/jquery-1.10.2.min.js")" type="text/javascript">
</script>
```

  - Ký tự ~: thư mục gốc của ứng dụng, trong ASP.NET MVC 5, sử dụng Razor 3, ký tự ~ được phân giải tự động khi nó xuất hiện trong thuộc tính src cho các phần tử script, style và img.

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Partial và RenderPartial

- **Html.Partial**: hiển thị một partial view thành dạng chuỗi, có 4 phương thức
  - public void Partial(string partialViewName);
  - public void Partial(string partialViewName, object model);
  - public void Partial(string partialViewName, ViewDataDictionary viewData);
  - public void Partial(string partialViewName, object model, ViewDataDictionary viewData);

```
@Html.Partial("AlbumDisplay")
```

- **Html.RenderPartial**: tương tự Partial nhưng ghi trực tiếp vào response nên nhanh hơn, nhưng phải đặt trong khối @{ }

```
@{ Html.RenderPartial("AlbumDisplay "); }
tương đương với:
@Html.Partial("AlbumDisplay ")
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Action và RenderAction

- Tương tự Partial và RenderPartial, nhưng linh hoạt hơn, ví dụ:

```
public class EmployeeController : Controller    {
        public ActionResult Index()
        {
            return View();
        }
        [ChildActionOnly]
        public ActionResult EmployeeLogin()
        {
            EmployeeLoginViewModel model = new EmployeeLoginViewModel();
            return PartialView("EmployeeLogin", model);
        }
        [ChildActionOnly]
        public ActionResult EmployeeRegistration()
        {
            EmployeeViewModel model = new EmployeeViewModel();
            return PartialView("EmployeeRegistration", model);
        }
}
```

Gọi trong View:

```
@Html.Action("EmployeeLogin")
...
@{Html.RenderAction("EmployeeRegistration");}
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.2 Form và các control

- Forms
- TextBox
- TextArea
- CheckBox
- RadioButton
- DropdownList
- Hidden
- Password
- String
- Label
- Editor

15

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Form

- **BeginForm()**: là một phương thức mở rộng cho cả hai lớp HtmlHelper và AjaxHelper để tạo phần tử <form>
- **Html.BeginForm()**:
  - Html.BeginForm("ActionMethod", "ControllerName","Get/Post Method")
    - ActionMethod: action khi form được submit.
    - ControllerName: tên Controller
    - Get/Post Method: phương thức được dùng để gởi dữ liệu từ form đến controller
      - Get: các tham số gởi đến máy chủ thông qua chuỗi truy vấn trong URL, thiếu tính bảo mật, không làm thay đổi trạng thái trên máy chủ nên được sử dụng trong các thao tác bình thường như tìm kiếm,...
      - Post: thông tin gởi về máy chủ nằm trong HTTP body, tính bảo mật cao, có thể làm thay đổi trạng thái trên máy chủ nên được sử dụng trong các trường hợp cần tính bảo mật như đăng nhập, gởi thông tin giao dịch, thêm vào giỏ hàng,...

16

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Form

- Ví dụ:

//Model: (**UserModel.cs**)

```
namespace HtmlHelperDemo.Models
{
  public class UserModel
  {
    public string UserName { get; set; }
    public int Age { get; set; }
    public string City { get; set; }
  }
}
```

//Controller: (HomeController.cs)

```
[HttpPost]
public ActionResult Index(UserModel user)
{
        return View(user);
}
```

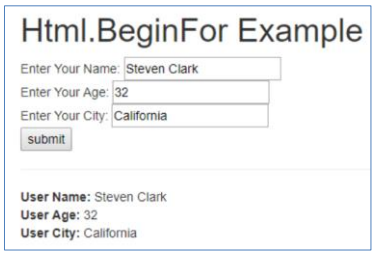

17

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Form

View:

```
@using HtmlHelperDemo.Models
@model UserModel
 <h1>Html.DisplayFor Example</h1>
@using (Html.BeginForm("Index", "Home", FormMethod.Post))
{
<span>Enter Your Name:</span> @Html.TextBoxFor(m => m.UserName)<br />
<span>Enter Your Age: </span> @Html.TextBoxFor(m => m.Age)<br />
<span>Enter Your City: </span>@Html.TextBoxFor(m => m.City)<br />
<input id = "Submit" type = "submit" value = "submit" />
} <hr />
<strong>User Name: </strong> @Html.DisplayFor(m => m.UserName)<br />
<strong>User Age: </strong> @Html.DisplayFor(m => m.Age)<br />
<strong>User City: </strong> @Html.DisplayFor(m => m.City)<br />
```



18

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - TextBox

- **Html.TextBox()**: tạo phần tử <input type="text" >
  - MvcHtmlString Html.TextBox(string name, string value, object htmlAttributes)

Model:
```
public class Student {
  public int StudentId { get; set; }
  [Display(Name="Name")]
  public string StudentName { get; set; }
  public int Age { get; set; }
  public bool isNewlyEnrolled { get; set; }
  public string Password { get; set; }
}
```

Mã HTML tương ứng:
```
<input class="form-control"
  id="StudentName"
  name="StudentName"
  type="text"
  value="" />
```

View:
```
@model Student
@Html.TextBox("StudentName", null, new { @class = "form-control" })
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper – TextBox (tt)

- Lưu ý:
  - Đối số đầu tiên trong Html.TextBox là tên một trường dữ liệu trong model
  - Nếu chỉ định một tên bất kỳ, textbox sẽ không gán kết với dữ liệu trong model

View:
```
@Html.TextBox("myTextBox", "This is value", new { @class = "form-control" })
```

Mã HTML tương ứng:
```
<input class="form-control"
  id="myTextBox"
  name="myTextBox"
  type="text"
  value="This is value" />
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper – TextBox (tt)

- **Html.TextBoxFor**: tạo một phần tử <input type="text"> sử dụng lambda expression
  - MvcHtmlString TextBoxFor(Expression<Func<TModel,TValue>> expression, object htmlAttributes)

View:
```
@model Student
@Html.TextBoxFor(m => m.StudentName, new { @class = "form-control" })
```

  - (giả sử StudentName= "John")

Mã HTML tương ứng:
```
<input class="form-control"
  id="StudentName"
  name="StudentName"
  type="text"
  value="John" />
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - TextArea

- **Html.TextArea()**: tạo phần tử <textarea rows="2" cols="20" >
  - MvcHtmlString Html.TextArea(string name, string value, object htmlAttributes)

Model:
```
public class Student {
  public int StudentId { get; set; }
  [Display(Name="Name")]
  public string StudentName { get; set; }
  public string Description { get; set; }
}
```

View:
```
@model Student
@Html.TextArea("Description", null, new { @class = "form-control" })
```

Mã HTML tương ứng:
```
<textarea class="form-control" id="Description" name="Description" rows="2" cols="20">
  This is value
</textarea>
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - TextArea

- **Html.TextAreaFor()**: tạo phần tử <textarea>, sử dụng lambda expression
  - MvcHtmlString TextAreaFor(<Expression<Func<TModel,TValue>> expression, object htmlAttributes)

View:
```
@model Student
@Html.TextAreaFor(m => m.Description, new { @class = "form-control" })
```

Mã HTML tương ứng:
```
<textarea class="form-control" id="Description" name="Description" rows="2" cols="20">
  This is value
</textarea>
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - CheckBox

- **Html.CheckBox()**: tạo phần tử <input type="checkbox" >
  - MvcHtmlString CheckBox(string name, bool isChecked, object htmlAttributes)

Model:
```
public class Student {
  public int StudentId { get; set; }
  [Display(Name="Name")]
  public string StudentName { get; set; }
  public int Age { get; set; }
  public bool isNewlyEnrolled { get; set; }
  public string Password { get; set; }
}
```

View:
```
@Html.CheckBox("isNewlyEnrolled", true)
```

Mã HTML tương ứng:
```
<input checked="checked"
  id="isNewlyEnrolled"
  name="isNewlyEnrolled"
  type="checkbox"
  value="true" />
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - CheckBox

- **Html.CheckBoxFor()**: tạo phần tử <input type="checkbox" >, sử dụng lambda expression
  - MvcHtmlString CheckBoxFor(<Expression<Func<TModel,TValue>> expression, object htmlAttributes)

View:
```
@model Student
@Html.CheckBoxFor(m => m.isNewlyEnrolled)
```

Mã HTML tương ứng:
```
<input data-val="true"
  data-val-required="The isNewlyEnrolled field is  required."
  id="isNewlyEnrolled"
  name="isNewlyEnrolled"
  type="checkbox" value="true" />
<input name="isNewlyEnrolled" type="hidden" value="false" />
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - RadioButton

- **Html.RadioButton()**: tạo phần tử <input type="radio" >
  - MvcHtmlString RadioButton(string name, object value, bool isChecked, object htmlAttributes)

Model:
```
public class Student {
  public int StudentId { get; set; }
  [Display(Name="Name")]
  public string StudentName { get; set; }
  public int Age { get; set; }
  public string Gender { get; set; }
}
```

View:
```
Male: @Html.RadioButton("Gender","Male")
Female: @Html.RadioButton("Gender","Female")
```

Mã HTML tương ứng:
```
Male: <input checked="checked" id="Gender" name="Gender" type="radio" value="Male" />
Female: <input id="Gender" name="Gender" type="radio" value="Female" />
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - RadioButton

- **Html.RadioButtonFor()**: tạo phần tử <input type="radio" >, sử dụng lambda expression
  - MvcHtmlString RadioButtonFor(<Expression<Func<TModel,TValue>> expression, object value, object htmlAttributes)

View:
```
@model Student
@Html.RadioButtonFor(m => m.Gender,"Male", new { @checked=true })
@Html.RadioButtonFor(m => m.Gender,"Female")
```

Mã HTML tương ứng:
```
<input checked="checked" id="Gender" name="Gender" type="radio" value="Male" />
<input id="Gender" name="Gender" type="radio" value="Female" />
```


27

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - DropDownList

- **Html.DropDownList()**: tạo phần tử <select>
  - MvcHtmlString Html.DropDownList(string name, IEnumerable<SelectLestItem> selectList, string optionLabel, object htmlAttributes)

Model:
```
public class Student {
  public int StudentId { get; set; }
  [Display(Name="Name")]
  public string StudentName { get; set; }
  public Gender StudentGender { get; set; }
}
public enum Gender { Male, Female }
```

View:
```
@using MyMVCApp.Models
@model Student
@Html.DropDownList("StudentGender",
    new SelectList(Enum.GetValues(typeof(Gender))),
    "Select Gender",
    new { @class = "form-control" })
```

Mã HTML tương ứng:
```
<select class="form-control" id="StudentGender" name="StudentGender">
  <option>Select Gender</option>
  <option>Male</option>
  <option>Female</option>
</select>
```


28

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - DropDownList

- **Html.DropDownList()**: dùng SelectList

Controller:

```
ViewBag.Student = new SelectList(db.StudentAddresses, "StudentID", "Address1", student.StudentID);
```

- db.StudentAddresses: Tên model
- StudentID: trường giá trị
- Address1: trường hiển thị
- student.StudentID: giá trị khi được chọn

View:

```
@Html.DropDownList("Address", (IEnumerable<SelectListItem>)ViewBag.Student,
                    htmlAttributes: new { @class = "form-control" })
```

- Address: thuộc tính name của phần tử <select> (select name="Address")
- (IEnumerable<SelectListItem>)ViewBag.StudentID: truy xuất dữ liệu từ ViewBag, ép về kiểu danh sách

29

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - DropDownList

- **Html.DropDownList()**: dùng danh sách bất kỳ

```
@{
  IEnumerable<string> strList = new List<string>{"BCA" , "BCS", "MCA", "MCS" };
}
@Html.DropDownList("ddlCourse",
                    new SelectList(strList,strList.FirstOrDefault()),
                   "--Select Course----" )
```

Mã HTML tương ứng:

```
<select id= "ddlCourse" name= "ddlCourse" >
      <option value= "" >--Select Course----</option>
      <option selected= "selected" >BCA</option>
      <option>BCS</option>
      <option>MCA</option>
      <option>MCS</option>
</select>
```


30

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - DropDownList

- **Html. DropDownListFor()**: tạo phần tử <select>, sử dụng lambda expression
  - MvcHtmlString Html.DropDownListFor(Expression<Func<dynamic,TProperty>> expression, IEnumerable<SelectLestItem> selectList, string optionLabel, object htmlAttributes)

View:
```
@using MyMVCApp.Models
@model Student
@Html.DropDownListFor(m => m.StudentGender,
                new SelectList(Enum.GetValues(typeof(Gender))),
                "Select Gender")
```

Mã HTML tương ứng:
```
<select class="form-control" id="StudentGender" name="StudentGender">
  <option>Select Gender</option>
  <option>Male</option>
  <option>Female</option>
</select>
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Hidden field

- **Html.Hidden()**: tạo phần tử <input type="hidden">
  - MvcHtmlString Html.Hidden(string name, object value, object htmlAttributes)

Model:
```
public class Student {
  public int StudentId { get; set; }
  [Display(Name="Name")]
  public string StudentName { get; set; }
  public int Age { get; set; }
  public bool isNewlyEnrolled { get; set; }
  public string Password { get; set; }
}
```

View:
```
@model Student
@Html.Hidden("StudentId")
```

Mã HTML tương ứng:
```
<input id="StudentId" name="StudentId" type="hidden" value="1" />
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Hidden field

- **Html. HiddenFor()**: tạo phần tử <input type="hidden">, sử dụng lambda expression
  - MvcHtmlString Html.HiddenFor(Expression<Func<dynamic,TProperty>> expression)

View:
```
@model Student
@Html.HiddenFor(m => m.StudentId)
```

Mã HTML tương ứng:
```
<input data-val="true"
  data-val-number="The field StudentId must be a number."
  data-val-required="The StudentId field is required."
  id="StudentId"
  name="StudentId"
  type="hidden"
  value="1" />
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Password

- **Html.Password()**: tạo phần tử <input type="password">
  - MvcHtmlString Html.Password(string name, object value, object htmlAttributes)

Model:
```
public class Student {
  public int StudentId { get; set; }
  [Display(Name="Name")]
  public string StudentName { get; set; }
  public int Age { get; set; }
  public bool isNewlyEnrolled { get; set; }
  public string OnlinePassword { get; set; }
}
```

View:
```
@model Student
@Html.Password("OnlinePassword")
```

Mã HTML tương ứng:
```
<input id="OnlinePassword" name="OnlinePassword" type="password" value="" />
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Password

- **Html. PasswordFor()**: tạo phần tử <input type="password">, sử dụng lambda expression
  - MvcHtmlString Html.PasswordFor(Expression<Func<dynamic,TProperty>> expression, object htmlAttributes)

View:
```
@model Student
@Html.PasswordFor(m => m.Password)
```

Mã HTML tương ứng:
```
<input id="Password" name="Password" type="password" value="mypassword" />
```


35

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Display HTML String

- **Html.Display()**: tạo một chuỗi (string) hiển thị trên trang web
  - MvcHtmlString Display(string expression)

Model:
```
public class Student {
  public int StudentId { get; set; }
  public string StudentName { get; set; }
  public int Age { get; set; }
}
```

View:
```
@Html.Display("StudentName")
```

- **Html.Display For()**: tạo một chuỗi (string) hiển thị trên trang web, sử dụng lambda expression

View:
```
@model Student
@Html.DisplayFor(m => m.StudentName)
```


36

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Label

- **Html.Label()**: tạo phần tử <label>
  - MvcHtmlString Label(string expression, string labelText, object htmlAttributes)

Model:
```
public class Student {
  public int StudentId { get; set; }
  public string StudentName { get; set; }
  public int Age { get; set; }
}
```

View:
```
@Html.Label("StudentName")
```

Mã HTML tương ứng:
```
<label for="StudentName">Name</label>
```

  - Có thể gán một chuỗi bất kỳ hiển thị trên Label:

View:
```
@Html.Label("StudentName","Student-Name")
```

Mã HTML tương ứng:
```
<label for="StudentName">Student-Name</label>
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Label

- **Html.LabelFor()**: tạo phần tử <label>, sử dụng lambda expression
  - MvcHtmlString LabelFor(<Expression<Func<TModel,TValue>> expression)

Model:
```
public class Student {
  public int StudentId { get; set; }
  public string StudentName { get; set; }
  public int Age { get; set; }
}
```

View:
```
@model Student
@Html.LabelFor(m => m.StudentName)
```

Mã HTML tương ứng:
```
<label for="StudentName">Name</label>
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Editor

- Tạo các phần tử html dựa trên kiểu dữ liệu của thuộc tính của đối tượng model

Property DataType	Html Element
string	<input type="text" >
int	<input type="number" >
decimal, float	<input type="text" >
boolean	<input type="checkbox" >
Enum	<input type="text" >
DateTime	<input type="datetime" >

39

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Editor

- **Html.Editor()**: yêu cầu tham số là một biểu thức chuỗi tên thuộc tính để tạo ra một phần tử html dựa trên kiểu dữ liệu của thuộc tính được chỉ định

Model:

```
public class Student {
   public int StudentId { get; set; }
   [Display(Name="Name")]
   public string StudentName { get; set; }
   public int Age { get; set; }
   public bool isNewlyEnrolled { get; set; }
   public string Password { get; set; }
   public string Gender { get; set; }
   public DateTime DoB { get; set; }
}
```

View:

```
StudentId: @Html.Editor("StudentId")
Student Name: @Html.Editor("StudentName")
Age: @Html.Editor("Age")
Password: @Html.Editor("Password")
isNewlyEnrolled: @Html.Editor("isNewlyEnrolled")
Gender: @Html.Editor("Gender")
DoB: @Html.Editor("DoB")
```

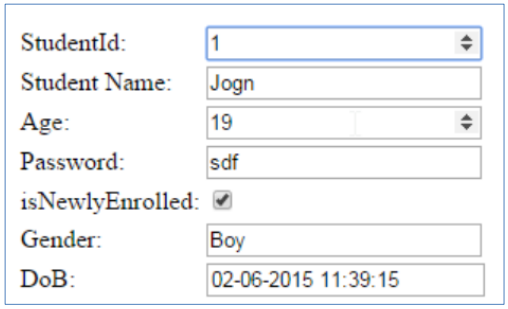

40

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

HtmlHelper - Editor

- **Html.EditorFor()**: Tạo phần tử HTML, sử dụng lambda expression

View:
StudentId: @Html.EditorFor(m => m.StudentId)
Student Name: @Html.EditorFor(m => m.StudentName)
Age: @Html.EditorFor(m => m.Age)
Password: @Html.EditorFor(m => m.Password)
isNewlyEnrolled: @Html.EditorFor(m => m.isNewlyEnrolled)
Gender: @Html.EditorFor(m => m.Gender)
DoB: @Html.EditorFor(m => m.DoB)



41

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Custom Helper

- Tạo HTML Helpers với Extension Methods: cho phép thêm các phương thức mới vào một lớp hiện có
  - Khai báo một class static
  - Phương thức mở rộng là một phương thức static, tham số đầu tiên được đặt trước từ khóa this, trường hợp này là this HtmlHelper. Ví dụ:

```
public static class ImageHelper {
    public static MvcHtmlString Image(this HtmlHelper helper, string src,
                                      string width, string height, string alt) {
        var builder = new TagBuilder("img");
        builder.MergeAttribute("src", src);
        builder.MergeAttribute("width", width);
        builder.MergeAttribute("height", height);
        builder.MergeAttribute("alt", alt);
        return MvcHtmlString.Create(builder.ToString(TagRenderMode.SelfClosing));
    }
}
```

Trong View: `@Html.Image("../images/rose.jpg","150","100","Error")`

42

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Custom Helper

- Tạo HTML Helpers với Extension Methods: cho phép thêm các phương thức mới vào một lớp hiện có
- Ví dụ: tạo một ImageHelper hiển thị hình ảnh:

```
public static class ImageHelper
{
        public static MvcHtmlString Image(this HtmlHelper helper, string src,
                                    string width, string height, string alt)
        {
            var builder = new TagBuilder("img");
            builder.MergeAttribute("src", src);
            builder.MergeAttribute("width", width);
            builder.MergeAttribute("height", height);
            builder.MergeAttribute("alt", alt);
            r                                                          ng));
        }
    }
```

Trong View:

```
@Html.Image("../images/rose.jpg","150","100","Error")
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3 Kiểm chứng dữ liệu

- Giới thiệu
- Required
- StringLength
- RegularExpression
- Range
- Compare
- Remote
- ...

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3.1 Giới thiệu về kiểm chứng dữ liệu

- Kiểm chứng dữ liệu phía client: dữ liệu từ client trước khi gởi đến server cần được kiểm tra tính hợp lệ của nó, nếu hợp lệ mới được gởi đi
- Kiểm chứng dữ liệu phía server: người dùng có thể vô hiệu hóa Javascript tìm cách vượt qua sự kiểm tra dữ liệu ở client. Do đó, cần thực hiện việc kiểm tra thông tin trên server.
- Data Annotations: tập hợp các thuộc tính và các class được định nghĩa trong System.ComponentModel.DataAnnotations, dùng để bổ sung thông tin cho class với metadata. MetaData là một bộ các quy tắc được dùng để chỉ ra đối tượng nào cần được kiểm tra.

45

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Giới thiệu về kiểm chứng dữ liệu (tt)

Các thuộc tính trong DataAnnotations

- Required
- Range
- RegularExpression,
- Compare
- StringLength
- Data type
- Credit Card number
- Currency
- Custom
- Date
- DateTime
- Duration
- Email Address
- HTML
- Image URL
- Multiline text
- Password
- Phone number
- Postal Code
- Upload

46

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Giới thiệu về kiểm chứng dữ liệu (tt)

- Các thuộc tính thường dùng trong kiểm chứng:
  - Compare: so sánh giá trị của hai thuộc tính trong Model, nếu bằng nhau, trả về true.
  - Remote: sử dụng JQuery Validation, gọi hàm trên Server thực hiện việc kiểm tra
  - Required: kiểm tra dữ liệu không được phép rỗng.
  - Range: kiểm tra dữ liệu số, được nhập vào thuộc phạm vi được chỉ ra
  - RegularExpression: dữ liệu được nhập vào được so khớp với biểu thức cho trước.
  - StringLength: giới hạn số ký tự được phép nhập vào.
  - …

47

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Giới thiệu về kiểm chứng dữ liệu (tt)

Attribute	Description
Required	Indicates that the property is a required field
StringLength	Defines a maximum length for string field
Range	Defines a maximum and minimum value for a numeric field
RegularExpression	Specifies that the field value must match with specified Regular Expression
CreditCard	Specifies that the specified field is a credit card number
CustomValidation	Specified custom validation method to validate the field
EmailAddress	Validates with email address format
FileExtension	Validates with file extension
MaxLength	Specifies maximum length for a string field
MinLength	Specifies minimum length for a string field
Phone	Specifies that the field is a phone number using regular expression for phone numbers

48

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Giới thiệu về kiểm chứng dữ liệu (tt)

- Hiển thị lỗi kiểm chứng
  - **ValidationMessage**
    - MvcHtmlString ValidateMessage(string modelName, string validationMessage, object htmlAttributes)

```
@model Student
@Html.Editor("StudentName") <br />
@Html.ValidationMessage("StudentName", "", new { @class = "text-danger" })
```

  - **ValidationMessageFor**
    - MvcHtmlString ValidateMessageFor(Expression<Func<dynamic,TProperty>> expression, string validationMessage, object htmlAttributes)

```
@model Student
@Html.EditorFor(m => m.StudentName) <br />
@Html.ValidationMessageFor(m => m.StudentName, "", new { @class = "text-danger" })
```

  - **ValidationSummary**
    - MvcHtmlString ValidateMessage(bool excludePropertyErrors, string message, object htmlAttributes)

```
@Html.ValidationSummary(false, "", new { @class = "text-danger" })
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Giới thiệu về kiểm chứng dữ liệu (tt)

- Các bước kiểm chứng dữ liệu :
  - Sử dụng các thuộc tính của lớp DataAnnotations, kết hợp với các trường dữ liệu cần kiểm chứng trong Model.

Required	CustomValidation	Phone
StringLength	EmailAddress	Display
Range	FileExtension	DataType
RegularExpression	MaxLength	...
CreditCard	MinLength	

  - Dùng các helper ValidationSummary, ValidationMessageFor hoặc ValidationMessage để hiển thị lỗi kiểm chứng cho từng field trong view.
  - Trong các Action, dùng ModelState.IsValid để kiểm tra tính hợp lệ của dữ liệu

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3.2 Required

- Bắt buộc nhập dữ liệu
- Bổ sung [Required] trước các thuộc tính cần kiểm chứng

Model:

```
[Required (ErrorMessage = "Please enter name")]
public string FirstName { get; set; }
[Required]
public string LastName { get; set; }
```

Controller:

```
[HttpPost]
public ActionResult Create(Student stud) {
        if (ModelState.IsValid)  {
            RedirectToAction("Index");
        }
        return View("index");
}
```

View:

```
@model ValidationDemo.Models.Student
...
@Html.EditorFor(model => model.FirstName, new { htmlAttributes = new { @class = "form-control" } })
@Html.ValidationMessageFor(model => model.FirstName, "", new { @class = "text-danger" })
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3.3 StringLength

- Giới hạn số ký tự

```
[Required]
[StringLength(160)]
public string FirstName { get; set; }
[Required]
[StringLength(160, MinimumLength=3)]
public string LastName { get; set; }
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3.4 RegularExpression

- Bắt buộc dữ liệu được nhập vào phải đúng qui tắc, ví dụ như email, ngày tháng, URL,...
- Cần cung cấp các biểu thức chính quy hợp lệ

```
[RegularExpression(@"[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}")]
public string Email { get; set; }
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

RegularExpression (tt)

Ký hiệu	Mô tả
a-z, A-Z	Ký tự chữ cái, đã được xác định, ví dụ: a
1-9	Ký tự số, đã được xác định, ví dụ: 1
[0-n]	Ký tự số từ 0 → n
[abc]	Một ký tự: a hoặc b hoặc c
|	Lựa chọn mẫu này hoặc mẫu khác(hoặc)
\w	Ký tự nhập là chữ cái, số và dấu gạch dưới
\W	kí tự nhập là khoảng trắng [\t\n\r]
\s	kí tự nhập phải khác khoảng trắng [^\t\n\r\f\v]

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

RegularExpressionValidator (tt)

Ký hiệu	Mô tả
\d	Ký tự nhập là ký tự số
\D	kí tự nhập phải khác kí tự số [^0-9]
\	Thể hiện ký tự đặc biệt theo sau
\.	Ký tự thay thế phải là dấu chấm
?	Quy định số lần xuất hiện: 0 hoặc 1 lần
*	Quy định số lần xuất hiện: 0 hoặc n lần
+	Số lần xuất hiện ít nhất 1 lần
{n}	Số lần xuất hiện đúng n lần
{n, m}	Số ký tự nằm trong khoảng n đến m lần

55

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

RegularExpressionValidator (tt)

- Ví dụ:
  [0-9]{3}-[0-9]{3}\s[0-9]{4}
  - [0-9] nhận tất cả các số 0-9
  - {3} yêu cầu 3 số được nhập cho phần đầu tiên
  - - là dấu bắt nhập
  - \s chỉ định một khoảng trắng

  ^([0-9]{10})$
  - Nhập đủ 10 số nguyên

56

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3.5 Range

- Kiểm tra dữ liệu kiểu số (mặc định là số nguyên), phải nằm trong giới hạn cho trước

```
[Range(35,44)]
public int Age { get; set; }
```

```
[Range(typeof(decimal), "0.00", "49.99")]
public decimal Price { get; set; }
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3.6 Compare

- Kiểm tra dữ liệu của hai thuộc tính trong model phải giống nhau

```
[EmailAddress]
public string Email { get; set; }
[Compare("Email")]
public string EmailConfirm { get; set; }
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3.7 Remote

- Thực hiện kiểm chứng dữ liệu phía client từ một lời gọi phía server

Controller

```
[Remote("CheckUserName", "Account")]
public string UserName { get; set; }
```

Client code:

```
public JsonResult CheckUserName(string username)
{
        var result = Membership.FindUsersByName(username).Count == 0;
        return Json(result, JsonRequestBehavior.AllowGet);
}
```


TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

4.3.8 Custom Validation

- Kiểm chứng theo qui định của người dùng
  - Tạo class kế thừa từ class ValidationAttribute
  - Khai báo trên thuộc tính cần kiểm chứng trong Model

```
public class CustomDate : ValidationAttribute {
    public override bool IsValid(object value)
    {
        DateTime dateTime = Convert.ToDateTime(value);
        return dateTime.Year < (DateTime.Now.Year – 18);
    }
}
```

```
[Required(ErrorMessage = "Please enter birthday")]
[Display(Name = "Birthday")]
[DisplayFormat(DataFormatString = "{0:d}", ApplyFormatInEditMode = true)]
[DataType (DataType.Date)]
[CustomDate(ErrorMessage = "Birthday must be less than current year - 18")]
public DateTime Birthday { get; set; }
```

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Ví dụ về kiểm chứng dữ liệu

Model:

```
public class Register
{
    [Display (Name="UserName")]
    [Required (ErrorMessage ="Please enter user name")]
    [StringLength (20, MinimumLength =3)]
    public string UserName { get; set; }
    [Display(Name ="Email Address")]
    [Required (ErrorMessage ="Please enter your email")]
    [RegularExpression (@"[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}", ErrorMessage ="Please enter valid email")]
    public string Email { get; set; }
    [Required (ErrorMessage ="Age is required")]
    [Range (1,100,ErrorMessage ="Please enter age between 1 and 100")]
    public int Age { get; set; }
    [Required (ErrorMessage ="Please enter password")]
    public string Password { get; set; }
    [Display (Name ="Confirm Password")]
    [Compare ("Password")]
    public string ConfirmPassword { get; set; }
}
```


61

TRƯỜNG ĐẠI HỌC MỞ TP. HCM
Cơ hội học tập cho mọi người

Ví dụ về kiểm chứng dữ liệu

View:

```
@using (@Html.BeginForm("Register", "Home", FormMethod.Post, null))
{
        @Html.LabelFor(m => m.UserName)
        @Html.TextBoxFor(m => m.UserName)
        @Html.ValidationMessageFor(m => m.UserName)<br />
        @Html.LabelFor(m => m.Age)
        @Html.TextBoxFor(m => m.Age)
        @Html.ValidationMessageFor(m => m.Age)<br />
        @Html.LabelFor(m => m.Email)
        @Html.TextBoxFor(m => m.Email)
        @Html.ValidationMessageFor(m => m.Email)<br />
        @Html.LabelFor(m => m.Password)
        @Html.TextBoxFor(m => m.Password)
        @Html.ValidationMessageFor(m => m.Password)<br />
        @Html.LabelFor(m => m.ConfirmPassword)
        @Html.TextBoxFor(m => m.ConfirmPassword)
        @Html.ValidationMessageFor(m => m.ConfirmPassword)<br />
       <input type="submit" value="Register" />
}
```

HomeController

```
[HttpPost]
    public ActionResult Register (Register reg)
    {
      if(ModelState.IsValid)
      {
        return RedirectToAction("Index","Home");
      }
      return View(reg);
    }
```

62