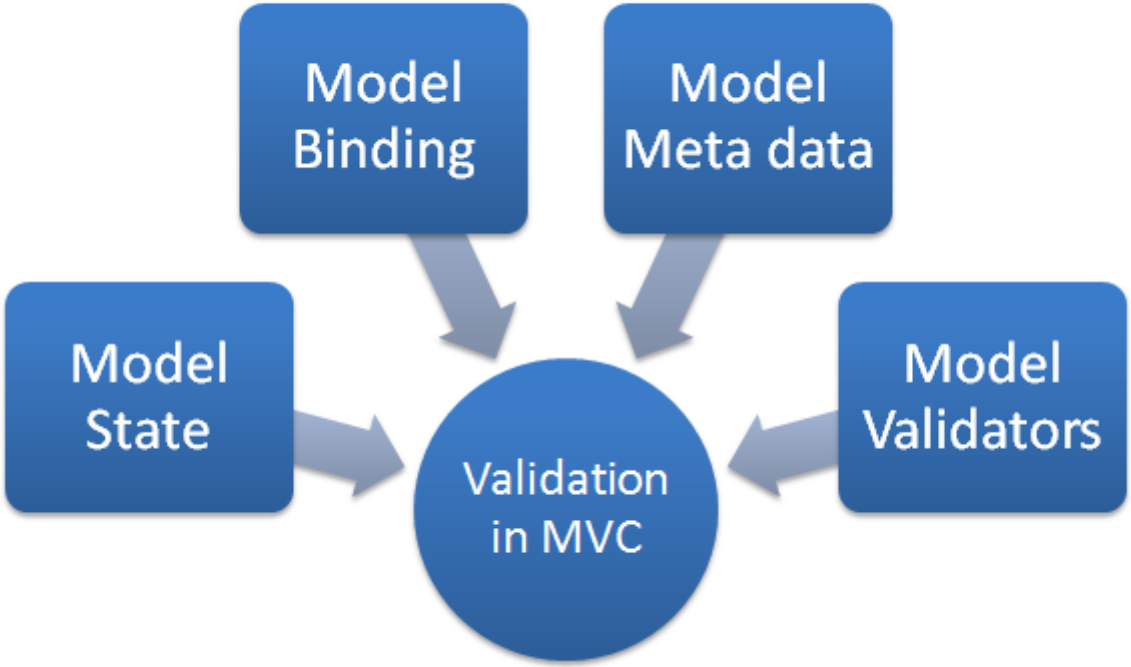
ASP.NET for Developer

GV: Bùi Quang Đăng

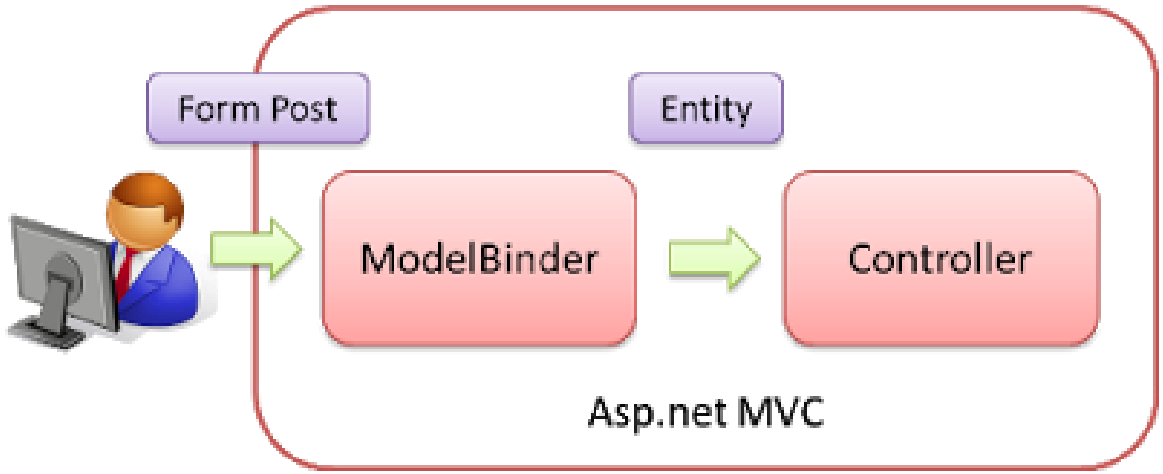
Contents


Model Binding
Model Meta data
Model State
Validation in MVC
Model Validators

Model Binding

Làm việc Model Binding

- **Model Binding** cho phép tạo ra một đối tượng trong .NET sử dụng để truyền gửi dữ liệu bởi trình duyệt trong một HTTP Request.



Làm việc với Model Binding

- **Model Binding** là cầu nối giữa HTTP Request và phương thức xác định hành động trong ASP.NET MVC.

Làm việc với Model Binding

- **Binding to Simple Types**
  - **Ví dụ**: Tạo 1 giao diện thêm mới phòng ban sau khi nhấn nút Create sẽ di chuyển và hiển thị thông tin nhập vào trên giao diện chi tiết
    - ThemMoi.aspx: Giao diện thêm mới phòng ban
    - PhongBanChiTiet.aspx: Giao diện hiển thị thông tin chi tiết
    - PhongBanController: Xử lý nghiệp vụ với thông tin phòng ban

Làm việc với Model Binding

- **Ví dụ**: Giao diện thêm mới

```
<% using (Html.BeginForm()) { %>
    <%: Html.AntiForgeryToken() %>
    <%: Html.ValidationSummary(true) %>

    <fieldset>
        <legend>PhongBan</legend>

        <div class="editor-label">
            <%: Html.LabelFor(model => model.MaPhong) %>
        </div>
        <div class="editor-field">
            <%: Html.EditorFor(model => model.MaPhong) %>
            <%: Html.ValidationMessageFor(model => model.MaPhong) %>
        </div>

        <div class="editor-label">
            <%: Html.LabelFor(model => model.TenPhong) %>
        </div>
        <div class="editor-field">
            <%: Html.EditorFor(model => model.TenPhong) %>
            <%: Html.ValidationMessageFor(model => model.TenPhong) %>
        </div>
        <p>
            <input type="submit" value="Create" />
        </p>
    </fieldset>
<% } %>
```

Làm việc với Model Binding

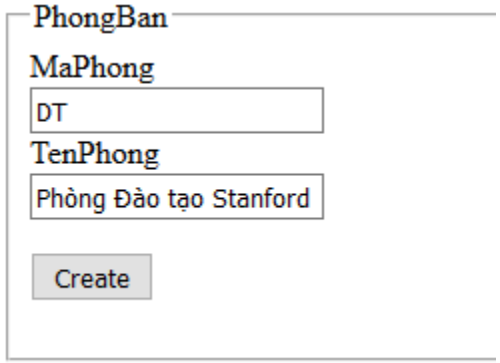
- **Ví dụ**: Giao diện hiển thị thông tin chi tiết

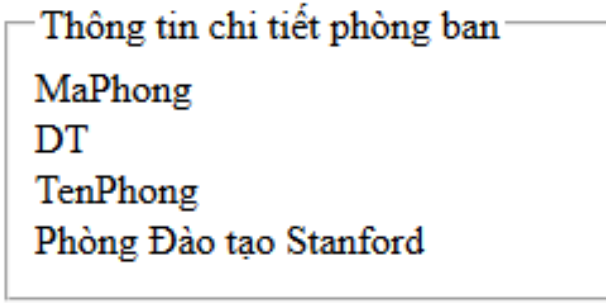
```
<fieldset>
    <legend>Thông tin chi tiết phòng ban</legend>

    <div class="display-label">
        <%: Html.DisplayNameFor(model => model.MaPhong) %>
    </div>
    <div class="display-field">
        <%: Html.DisplayFor(model => model.MaPhong) %>
    </div>

    <div class="display-label">
        <%: Html.DisplayNameFor(model => model.TenPhong) %>
    </div>
    <div class="display-field">
        <%: Html.DisplayFor(model => model.TenPhong) %>
    </div>
</fieldset>
```

Làm việc với Model Binding

- **Ví dụ**: Phương thức xử lý trong PhongBanController, giao diện hiển thị

```
0 references
public ActionResult ThemMoi()
{
    return View();
}

[HttpPost]
0 references
public ActionResult ThemMoi(PhongBan objPhong)
{
    return View("PhongBanChiTiet", objPhong);
}
```





Làm việc với Model Binding

- **Binding to Complex Types**

```
20 references
public class NhanVien
{
    3 references
    public NhanVien() {

    }
    3 references
    public string MaNV { get; set; }

    3 references
    public string HoTen { get; set; }

    0 references
    public PhongBan PB { get; set; }
}
```

```
11 references
public class PhongBan
{
    3 references
    public PhongBan()
    {

    }
    3 references
    public string MaPhong { get; set; }

    3 references
    public string TenPhong { get; set; }
}
```

Làm việc với Model Binding

Binding to Complex Types

- Giao diện thêm mới phòng ban:

```
<% using (Html.BeginForm("HienThiPhong", "NhanVien")) { %>
    <%: Html.AntiForgeryToken() %>
    <%: Html.ValidationSummary(true) %>

    <fieldset>
        <legend>NhanVien</legend>

        <div class="editor-label">
            <%: Html.LabelFor(model => model.MaNV) %>
        </div>
        <div class="editor-field">
            <%: Html.EditorFor(model => model.MaNV) %>
            <%: Html.ValidationMessageFor(model => model.MaNV) %>
        </div>
```

Làm việc với Model Binding

Binding to Complex Types

- Giao diện thêm mới phòng ban:

```
<div class="editor-label">
    <%: Html.LabelFor(model => model.PB.TenPhong) %>
</div>
<div class="editor-field">
    <%: Html.EditorFor(model => model.PB.TenPhong) %>
    <%: Html.ValidationMessageFor(model => model.PB.TenPhong) %>
</div>

<p>
    <input type="submit" value="Create" />
</p>
```

Làm việc với Model Binding

Binding to Complex Types

- Xử lý trong NhanVienController sử dụng thuộc tính Bind trong phương thức:

```
0 references
public ActionResult ThemMoi()
{
    PhongBanBusiness s = new PhongBanBusiness();

    ViewBag.ddlPhongBan = new SelectList(s.GetListPhongBan(), "MaPhong", "TenPhong");

    return View();
}

0 references
public ActionResult HienThiPhong([Bind(Prefix="PB")] PhongBan phong)
{
    return View(phong);
}
```

Model Validation

Giới thiệu về Model Validation

- Là kỹ thuật cho phép kiểm tra, bắt lỗi (validation) các đối tượng, thành phần thông tin trên giao diện web viết bằng MVC
- Người lập trình có thể sử dụng các cách thức sau:
  - **Explicitly Validating a Model**
  - **Model Validation using Data Annotations**

Explicitly Validating a Model

Explicitly Validating a Model

- Là cách thức kiểm tra, bắt lỗi thông qua đối tượng **ModelState** trong MVC.
- Ưu điểm của cách thức này là tùy biến và linh động theo cách sử dụng của người lập trình thông qua phương thức **AddModelError** để xác định thuộc tính, nội dung muốn thông báo.

Explicitly Validating a Model

- **Ví dụ:**

```
if (string.IsNullOrEmpty(objCag.CategoryName))
            {
                ModelState.AddModelError("CategoryName", "Bạn
cần phải nhập tên chủ đề");
            }
```

- Sử dụng thuộc tính **ModelState.IsValid** để thực hiện công việc sau khi các thông tin cần kiểm tra, bắt lỗi (validation) đạt yêu cầu.

Explicitly Validating a Model

- **Ví dụ:**

```
if (ModelState.IsValid)
        {
            Common.Entities.stanfCategories.Add(objCag);

            Common.Entities.SaveChanges();

            return RedirectToAction("Index");
        }
```

Displaying Validation Messages

- Sử dụng để hiển thị thông báo trong quá trình kiểm tra, bắt lỗi ra giao diện người dùng trong MVC.
- Hàm hiển thị các thông báo lỗi trên giao diện **ValidationSummary()**
- Hàm hiển thị theo từng thông tin cần kiểm tra, bắt lỗi **ValidationMessageFor()**

Displaying Validation Messages

- **Ví dụ:**

```
<div class="form-group">
        @Html.LabelFor(model => model.CategoryName,
htmlAttributes: new { @class = "control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.CategoryName, new
{ htmlAttributes = new { @class = "form-control" } })
            @Html.ValidationMessageFor(model =>
model.CategoryName, "", new { @class = "text-danger" })
</div>
```

Model Validation using Data Annotations

Validation Rules Using Metadata

```
0 references
public int Id { get; set; }

[Display(Name="Tên chủ đề")]
[Required(ErrorMessage="Bạn cần phải nhập chủ đề bài viết")]
[StringLength(50, MinimumLength=10, ErrorMessage="Bạn cần nhập tối thiểu {2} kí tự")]
0 references
public string CategoryName { get; set; }

[Display(Name = "Mô tả")]
0 references
public string CategoryDescription { get; set; }
```

Validation Rules Using Metadata

```
<div class="form-group">
    @Html.LabelFor(model => model.CategoryDescription, htmlAttributes:
      new { @class = "control-label col-md-2" })
    <div class="col-md-10">
        @Html.EditorFor(model => model.CategoryDescription, new
          { htmlAttributes = new { @class = "form-control" } })
        @Html.ValidationMessageFor(model => model.CategoryDescription,
          "", new { @class = "text-danger" })
    </div>
</div>
<div class="form-group">
    <div class="col-md-offset-2 col-md-10">
        <input type="submit" value="Tạo mới" class="btn btn-default" />
    </div>
</div>
```

Validation Rules Using Metadata

- Sử dụng thuộc tính để kiểm tra, bắt lỗi cần khai báo thư viện **System.ComponentModel.DataAnnotations**
- Ưu điểm của cách thức này là người lập trình không phải viết code, kết hợp với javascript để validation phía client.

Validation Rules Using Metadata

- **Ví dụ:**

```
[Display(Name="Tên chủ đề")]
[Required(ErrorMessage="Bạn cần phải nhập chủ đề bài viết")]

public string CategoryName { get; set; }
```

Validation Rules Using Metadata

- **Validation Attributes:**

Attribute	Description
[Required]	Yêu cầu người dùng cần phải nhập thông tin
[StringLength]	Xác định độ dài tối đa của trường thông tin, ví dụ: [StringLength(10)]
[Range]	Giá trị nhập trong khoảng cho phép, ví dụ: [Range(10, 20)]
[DisplayName]	Hiển thị tên của thuộc tính, ví dụ [DisplayName("Chủ đề")]

Validation Rules Using Metadata

- **Validation Attributes:**

Attribute	Description
[DisplayFormat]	Sử dụng để hiển thị định dạng, ví dụ: [DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}", ApplyFormatInEditMode = true)]
[Compare]	So sánh với trường khác, ví dụ: [Compare("Password Retype")]
[DataType]	Xác định kiểu của trường thông tin, ví dụ: [DataType(DataType.Date)]

Validation Rules Using Metadata

- **Validation Attributes:**

Attribute	Description
MaxLengthAttribute	Độ dài tối đa của trường thông tin
MinLengthAttribute	Độ dài tối thiểu của trường thông tin
RegularExpressionAttribute	Kiểm tra theo biểu thức chính quy, ví dụ: [DataType(DataType.PhoneNumber)] [RegularExpression("^[0-9]{8}$")] public string Phone { get; set; }

Validation Rules Using Metadata

- **Ví dụ:**

```
[DisplayName("Ngày sinh")]
[DataType(DataType.Date)]
[DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}",
ApplyFormatInEditMode = true)]

public DateTime? Birth { get; set; }
```

```
[Display(Name="Tên chủ đề")]
[Required(ErrorMessage="Bạn cần phải nhập chủ đề bài viết")]
[StringLength(50, MinimumLength=10, ErrorMessage="Bạn cần
nhập tối thiểu {2} kí tự")]

public string CategoryName { get; set; }
```

Client-Side Validation

Client-Side Validation

- Là cách thức sử dụng thư viện javascript để kiểm tra, bắt lỗi từ client trong MVC
- Sử dụng khai báo trong web.config như sau:
  - **<add key="ClientValidationEnabled" value="true" />**
  - **<add key="UnobtrusiveJavaScriptEnabled" value="true" />**

Client-Side Validation

- Thực hiện cài đặt các thư viện cần thiết nếu chưa có bằng NuGet bằng cách vào Visual Studio chọn **Tools => Library Package Manager** sau đó khai báo:
  - Install-Package jQuery –version 1.10.2
  - Install-Package jQuery.Validation –version 1.11.1
  - Install-Package Microsoft.jQuery.Unobtrusive.Validation –version 3.0.0

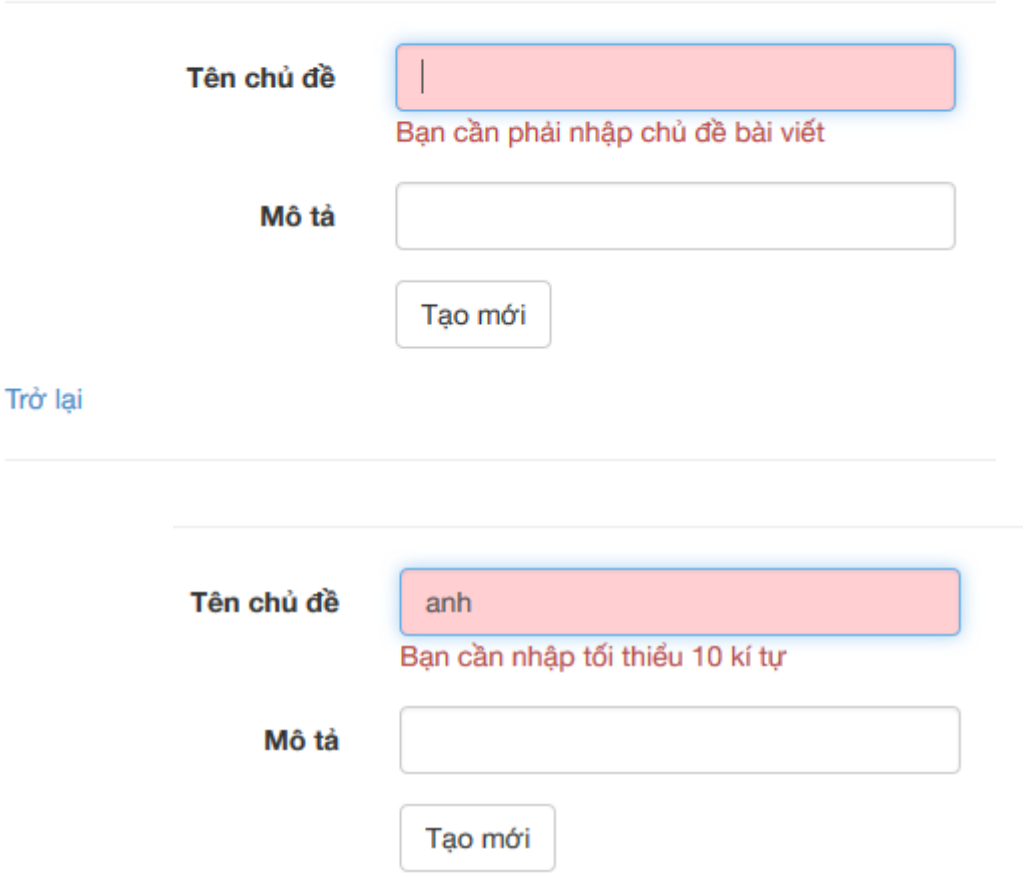
Client-Side Validation

- Khai báo các thư viện để sử dụng trên trang web:

```
<script src="~/Scripts/jquery-1.10.2.js"></script>
<script src="~/Scripts/jquery.validate.js"></script>
<script src="~/Scripts/jquery.validate.unobtrusive.js"></script>
```

```
[Display(Name="Tên chủ đề")]
[Required(ErrorMessage="Bạn cần phải nhập chủ đề bài viết")]
[StringLength(50, MinimumLength=10, ErrorMessage="Bạn cần
nhập tối thiểu {2} kí tự")]

public string CategoryName { get; set; }
```

Client-Side Validation



Practices


Stanford
nâng tầm tri thức
Thank You !
www.stanford.com.vn