MVC Validation

<https://github.com/JeremySkinner/FluentValidation/wiki/h.-ASP.NET-MVC-5-integration>

Global.asax

Insert:

FluentValidationModelValidatorProvider.Configure();

Stage2Validator.cs

Build the Validator:

using Mikes.Models;

using FluentValidation;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace MikesScreens.Validators

{

public class Stage2Validator:AbstractValidator<Stage2>

{

public Stage2Validator()

{

RuleFor(stage2 => stage2.Title)

.NotNull()

.WithMessage("Please specify a title Mike");

}

}

}

//--------------------------------------------------------------------------------------------

Stage2.cs

Insert the two libraries in red, validator method, and blank out the first two lines:

using FluentValidation.Attributes;

using MikesScreens.Validators;

namespace Mikes.Models

{

[Validator(typeof(Stage2Validator))]

public class Stage2

{

//[Required(ErrorMessage = "Title Is Required")]

//[MinLength(2, ErrorMessage = "At Least 2 Letters Are Required")]

[Display(Name = "Title", Prompt = "Please Supply Title", Description = "Title")]

public string Title { get; set; }

//--------------------------------------------------------------------------------------------

Stage2.cshtml

Insert at the top: @Html.ValidationSummary()

Delete: @Html.AntiForgeryToken()

Delete all old validation messages:

@\*@Html.ValidationMessageFor(model => model.ClientTypeIndividualOrCompany, "", new { @class = "text-danger" })\*@

Delete the old “Next” button and create a new one, don’t do the “Back” button:

@\*<button type="button" onclick="location.href='@Url.Action("Step3")'">Next</button>\*@

<input type="submit" value="Step3" />

//--------------------------------------------------------------------------------------------

Controller.cs

Build the “Set()”

// GET: Step 2---------------------------

public ActionResult Step2()

{

return View();

}

// SET: Step 2.

[HttpPost]

public ActionResult Step2(Stage2 stage2)

{

if (!ModelState.IsValid)

{

// re-render the view when validation failed.

return View("Step2", stage2);

}

return RedirectToAction("Step3");

}