عنوان: **استفاده از FluentValidation در ASP.NET MVC**
نویسنده: محمّد زارع
تاریخ: ۱۳۹۱/۰۸/۲۰ ۱۶:۵
آدرس: www.dotnettips.info
برچسب‌ها: MVC, Validation, FluentValidation

برای هماهنگی این کتابخانه با ASP.NET MVC نیاز به نصب FluentValidation.Mvc3 یا FluentValidation.Mvc4 از طریق Nuget یا دانلود کتابخانه از سایت CodePlex می‌باشد. بعد از نصب کتابخانه، نیاز به تنظیم FluentValidationModelValidatorProvider داخل متد Application_Start (فایل Global.asax) داریم:

```
protected void Application_Start() {
    AreaRegistration.RegisterAllAreas();

    RegisterGlobalFilters(GlobalFilters.Filters);
    RegisterRoutes(RouteTable.Routes);

    FluentValidationModelValidatorProvider.Configure();
}
```

تصور کنید دو کلاس Person و PersonValidator را به صورت زیر داریم:

```
[Validator(typeof(PersonValidator))]
    public class Person {
    public int Id { get; set; }
    public string Name { get; set; }
    public string Email { get; set; }
    public int Age { get; set; }
}

public class PersonValidator : AbstractValidator<Person> {
    public PersonValidator() {
        RuleFor(x => x.Id).NotNull();
        RuleFor(x => x.Name).Length(0, 10);
        RuleFor(x => x.Email).EmailAddress();
        RuleFor(x => x.Age).InclusiveBetween(18, 60);
    }
}
```

همان طور که ملاحظه می‌کنید، در بالای تعریف کلاس Person با استفاده از ValidatorAttribute مشخص کرده ایم که از PersonValidator جهت اعتبارسنجی استفاده کند.
در آخر می‌توانیم Controller و View ی برنامه مان را درست کنیم:

```
public class PeopleController : Controller {
    public ActionResult Create() {
        return View();
    }

    [HttpPost]
    public ActionResult Create(Person person) {

        if(! ModelState.IsValid) { // re-render the view when validation failed.
            return View("Create", person);
        }

        TempData["notice"] = "Person successfully created";
        return RedirectToAction("Index");
    }
}
```

```
@Html.ValidationSummary()

@using (Html.BeginForm()) {
Id: @Html.TextBoxFor(x => x.Id) @Html.ValidationMessageFor(x => x.Id)
<br />
Name: @Html.TextBoxFor(x => x.Name) @Html.ValidationMessageFor(x => x.Name)
<br />
Email: @Html.TextBoxFor(x => x.Email) @Html.ValidationMessageFor(x => x.Email)
<br />
Age: @Html.TextBoxFor(x => x.Age) @Html.ValidationMessageFor(x => x.Age)
```

```
<input type="submit" value="submit" />
}
```

اکنون DefaultModelBinder موجود در MVC برای اعتبارسنجی شیء Person از FluentValidationModelValidatorProvider استفاده خواهد کرد.
توجه داشته باشید که FluentValidation با اعتبارسنجی سمت کاربر ASP.NET MVC به خوبی کار خواهد کرد منتها نه برای تمامی اعتبارسنجی ها. به عنوان مثال تمام قوانینی که از شرط‌های When/Unless استفاده کرده اند، Validatorهای سفارشی، و قوانینی که در آن‌ها از Must استفاده شده باشد، اعتبارسنجی سمت کاربر نخواهند داشت. در زیر لیست Validator هایی که با اعتبارسنجی سمت کاربر به خوبی کار خواهند کرد آمده است:

NotNull/NotEmpty

Matches

InclusiveBetween

CreditCard

Email

EqualTo

Length

صفت CustomizeValidator

با استفاده از CustomizeValidatorAttribute می‌توان نحوه اجرای Validator را تنظیم کرد. به عنوان مثال اگر میخواهید که Validator تنها برای یک RuleSet مخصوص انجام شود می‌توانید مانند زیر عمل کنید:

```
public ActionResult Save([CustomizeValidator(RuleSet="MyRuleset")] Customer cust) {
  // ...
}
```

مواردی که تا اینجا گفته شد برای استفاده در یک برنامه‌ی ساده MVC کافی به نظر می‌رسد، اما از اینجا به بعد مربوط به مواقعی است که نخواهیم از Attributeها استفاده کنیم و کار را به IoC بسپاریم.

استفاده از Validator Factory با استفاده از یک IoC Container

Validator Factory چیست؟ Validator Factory یک کلاس می‌باشد که وظیفه ساخت نمونه از Validatorها را بر عهده دارد. اینترفیس IValidatorFactory به صورت زیر می‌باشد:

```
public interface IValidatorFactory {
   IValidator<T> GetValidator<T>();
   IValidator GetValidator(Type type);
}
```

ساخت Validator Factory سفارشی:

برای ساخت یک Validator Factory شما می‌توانید به طور مستقیم IValidatorFactory را پیاده سازی نمایید یا از کلاس ValidatorFactoryBase به عنوان کلاس پایه استفاده کنید (که مقداری از کارها را برای شما انجام داده است). در این مثال نحوه ایجاد یک Validator Factory که از StructureMap استفاده می‌کند را بررسی خواهیم کرد. ابتدا نیاز به ثبت Validatorها در

StructureMap داریم:

```
ObjectFactory.Configure(cfg => cfg.AddRegistry(new MyRegistry()));

public class MyRegistry : Registry {
    public MyRegistry() {
        For<IValidator<Person>>()
    .Singleton()
    .Use<PersonValidator>();
    }
}
```

در اینجا StructureMap را طوری تنظیم کرده ایم که از یک Registry سفارشی استفاده کند. در داخل این Registry به StructureMap میگوییم که زمانی که خواسته شد تا یک نمونه از IValidator<Person> ایجاد کند، PersonValidator را بر گرداند. متد CreateInstance نوع مناسب را نمونه سازی می‌کند (CustomerValidator) و آن را بازمی گرداند (یا Null بر می‌گرداند اگر نوع مناسبی وجود نداشته باشد)

استفاده از AssemblyScanner

FluentValidation دارای یک AssemblyScanner می‌باشد که کار ثبت Validatorها داخل یک اسمبلی را راحت‌تر می‌سازد. با استفاده از AssemblyScanner کلاس MyRegistery ما شبیه قطعه کد زیر خواهد شد:

```
public class MyRegistry : Registry {
   public MyRegistry() {

     AssemblyScanner.FindValidatorsInAssemblyContaining<MyValidator>()
       .ForEach(result => {
            For(result.InterfaceType)
               .Singleton()
               .Use(result.ValidatorType);
       });
   }
}
```

حالا زمان استفاده از factory ساخته شده در متد Application_Start برنامه می‌باشد:

```
protected void Application_Start() {
    RegisterRoutes(RouteTable.Routes);

    //Configure structuremap
    ObjectFactory.Configure(cfg => cfg.AddRegistry(new MyRegistry()));
    ControllerBuilder.Current.SetControllerFactory(new StructureMapControllerFactory());

    //Configure FV to use StructureMap
    var factory = new StructureMapValidatorFactory();

    //Tell MVC to use FV for validation
    ModelValidatorProviders.Providers.Add(new FluentValidationModelValidatorProvider(factory));
    DataAnnotationsModelValidatorProvider.AddImplicitRequiredAttributeForValueTypes = false;
}
```

اکنون FluentValidation از StructureMap برای نمونه سازی Validatorها استفاده خواهد کرد و کار اعتبارسنجی مدل‌ها به FluentValidaion سپرده شده است.

نظرات خوانندگان

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۸/۲۱ ۸:۵۱

با تشکر مجدد از شما...
برای مواردی که سمت کلاینت ساپورت نمیشن راه حل ی وجود داره؟
بهتره این اعتبارسنجی‌ها تو کدوم لایه نوشته بشن؟

نویسنده: میثم زارع
تاریخ: ۱۳۹۱/۰۸/۲۱ ۹:۴۷

در مورد سوال اول : Custom validators with client side و Enable custom validator client side in FV
در مورد سوال دوم هم، اکثر مواقع روی ViewModel یا بهتر بگم InputModel انجام میشه، هر چند اگر نیاز بود میشه روی خود کلاس Entity مورد نظر هم ایجاد کرد.
اینجا خود سازنده کتابخانه توضیح داده که چطور ازش استفاده میکنه:
http://fluentvalidation.codeplex.com/discussions/355068

نویسنده: نارینه
تاریخ: ۱۳۹۱/۱۱/۰۱ ۱۳:۲۶

StructureMapValidatorFactory که در Application_Start() استفاده شده است، د رکجا و به چه شکلی تعریف گردیده؟
امکان دارد نمونه کد کامل را جهت استفاده قرار دهید؟

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۱۱/۰۱ ۱۴:۳۷

```
public class StructureMapValidatorFactory : ValidatorFactoryBase {
public override IValidator CreateInstance(Type validatorType) {
return ObjectFactory.TryGetInstance(validatorType) as IValidator;
}
}
```

اطلاعات بیشتر