عنوان: **MVC Scaffolding #2**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۰۳ ۸:۱۰
آدرس: www.dotnettips.info
برچسب‌ها: Entity framework, MVC, MVC Scaffolding

از آنجائیکه اصل کار با MVC Scaffolding از طریق خط فرمان پاورشل انجام می‌شود، بنابراین بهتر است در ادامه با گزینه‌ها و سوئیچ‌های مرتبط با آن بیشتر آشنا شویم.
دو نوع پارامتر حین کار با MVC Scaffolding مهیا هستند:

الف) سوئیچ‌ها
مانند پارامترهای boolean عمل کرده و شامل موارد ذیل می‌باشند. تمام این پارامترها به صورت پیش فرض دارای مقدار false بوده و ذکر هرکدام در دستور نهایی سبب true شدن مقدار آن‌ها می‌گردد:
Repository: برای تولید کدها بر اساس الگوی مخزن
Force: برای بازنویسی فایل‌های موجود.
ReferenceScriptLibraries: ارجاعاتی را به اسکریپت‌های موجود در پوشه Scripts، اضافه می‌کند.
NoChildItems: در این حالت فقط کلاس کنترلر تولید می‌شود و از سایر ملحقات مانند تولید Viewها، DbContext و غیره صرفنظر خواهد شد.

ب) رشته‌ها
این نوع پارامترها، رشته‌ای را به عنوان ورودی خود دریافت می‌کنند و شامل موارد ذیل هستند:
ControllerName: جهت مشخص سازی نام کنترلر مورد نظر
ModelType: برای ذکر صریح کلاس مورد استفاده در تشکیل کنترلر بکار می‌رود. اگر ذکر نشود، از نام کنترلر حدس زده خواهد شد.
DbContext: نام کلاس DbContext تولیدی را مشخص می‌کند. اگر ذکر نشود از نامی مانند ProjectNameContex استفاده خواهد کرد.
Project: پیش فرض آن پروژه جاری است یا اینکه می‌توان پروژه دیگری را برای قرار دادن فایل‌های تولیدی مشخص کرد. (برای مثال هربار یک سری کد مقدماتی را در یک پروژه جانبی تولید کرد و سپس موارد مورد نیاز را از آن به پروژه اصلی افزود)
CodeLanguage: می‌تواند cs یا vb باشد. پیش فرض آن زبان جاری پروژه است.
Area: اگر می‌خواهید کدهای تولیدی در یک ASP.NET MVC area مشخص قرار گیرند، نام Area مشخصی را در اینجا ذکر کنید.
Layout: در حالت پیش فرض از فایل layout اصلی استفاده خواهد شد. اما اگر نیاز است از layout دیگری استفاده شود، مسیر نسبی کامل آن‌را در اینجا قید نمائید.

یک نکته:
نیازی به حفظ کردن هیچکدام از موارد فوق نیست. برای مثال در خط فرمان پاورشل، دستور Scaffold را نوشته و پس از یک فاصله، دکمه Tab را فشار دهید. لیست پارامترهای قابل اجرای در این حالت ظاهر خواهند شد. اگر در اینجا برای نمونه Controller انتخاب شود، مجددا با ورود یک فاصله و خط تیره و سپس فشردن دکمه Tab، لیست پارامترهای مجاز و همراه با سوئیچ کنترلر ظاهر می‌گردند.

MVC Scaffolding و مدیریت روابط بین کلاس‌ها

مثال قسمت قبلی بسیار ساده و شامل یک کلاس بود. اگر آن‌را کمی پیچیده‌تر کرده و برای مثال روابط one-to-many و many-to-many را اضافه کنیم چطور؟

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
```

```
namespace MvcApplication1.Models
{
    public class Task
    {
        public int Id { set; get; }

        [Required]
        public string Name { set; get; }

        [DisplayName("Due Date")]
        public DateTime? DueDate { set; get; }

        [ForeignKey("StatusId")]
        public virtual Status Status { set; get; } // one-to-many
        public int StatusId { set; get; }

        [StringLength(450)]
        public string Description { set; get; }

        public virtual ICollection<Tag> Tags { set; get; } // many-to-many
    }

    public class Tag
    {
        public int Id { set; get; }

        [Required]
        public string Name { set; get; }

        public virtual ICollection<Task> Tasks { set; get; } // many-to-many
    }

    public class Status
    {
        public int Id { set; get; }

        [Required]
        public string Name { set; get; }
    }
}
```

کلاس Task تعریف شده اینبار دارای رابطه many-to-many با برچسب‌های مرتبط با آن است. همچنین یک رابطه one-to-many با کلاس وضعیت هر Task نیز تعریف شده است. به علاوه نکته تعریف « کار با کلیدهای اصلی و خارجی در EF Code first » نیز در اینجا لحاظ گردیده است.

در ادامه دستور تولید کنترلرهای Task، Tag و Status ساخته شده با الگوی مخزن را در خط فرمان پاورشل ویژوال استودیو صادر می‌کنیم:

```
PM> Scaffold Controller -ModelType Task -ControllerName TasksController -DbContextType TasksDbContext -
Repository -Force
PM> Scaffold Controller -ModelType Tag -ControllerName TagsController -DbContextType TasksDbContext -
Repository -Force
PM> Scaffold Controller -ModelType Status -ControllerName StatusController -DbContextType
TasksDbContext -Repository -Force
```

اگر به کارهایی که در اینجا انجام می‌شود دقت کنیم، می‌توان صرفه جویی زمانی قابل توجهی را شاهد بود؛ خصوصا در برنامه‌هایی که از ده‌ها فرم ورود اطلاعات تشکیل شده‌اند. فرض کنید قصد استفاده از ابزار فوق را نداشته باشیم. باید به ازای هر عملیات CRUD دو متد را ایجاد کنیم. یکی برای نمایش و دیگری برای ثبت. بعد بر روی هر متد کلیک راست کرده و Viewهای متناظری را ایجاد کنیم. سپس مجددا یک سری پیاده سازی «مقدماتی» تکراری را به ازای هر متد جهت ثبت یا ذخیره اطلاعات تدارک ببینیم.

اما در اینجا پس از طراحی کلاس‌های برنامه، با یک دستور، حجم قابل توجهی از کدهای «مقدماتی» که بعدها مطابق نیاز ما سفارشی سازی و غنی‌تر خواهند شد، تولید می‌گردند.

چند نکته:

- با توجه به اینکه مدل‌ها تغییر کرده‌اند، نیاز است بانک اطلاعاتی متناظر نیز به روز گردد. مطالب مرتبط با آن‌را در مباحث Migrations می‌توانید مطالعه نمائید.

- View تولیدی رابطه many-to-many را پشتیبانی نمی‌کند. این مورد را باید دستی اضافه و طراحی کنید: (^ و ^)

- رابطه one-to-many به خوبی با View متناظری دارای یک drop down list تولید خواهد شد. در اینجا لیست تولیدی به صورت خودکار با مقادیر خاصیت Name کلاس Status پر می‌شود. اگر این نام دقیقا Name نباشد نیاز است توسط ویژگی به نام DisplayColumn که بر روی نام کلاس قرار می‌گیرد، مشخص کنید از کدام خاصیت باید استفاده شود.

```
@Html.DropDownListFor(model => model.StatusId,
((IEnumerable<Status>)ViewBag.PossibleStatus).Select(option => new SelectListItem {
  Text = (option == null ? "None" : option.Name),
  Value = option.Id.ToString(),
  Selected = (Model != null) && (option.Id == Model.StatusId)
}), "Choose...")
@Html.ValidationMessageFor(model => model.StatusId)
```

تولید آزمون‌های واحد به کمک MVC Scaffolding

MVC Scaffolding امکان تولید خودکار کلاس‌ها و متدهای آزمون واحد را نیز دارد. برای این منظور دستور زیر را در خط فرمان پاورشل وارد نمائید:

```
PM> Scaffold MvcScaffolding.ActionWithUnitTest -Controller TasksController -Action ArchiveTask -
ViewModel Task
```

دستوری که در اینجا صادر شده است نسبت به حالت‌های کلی قبلی، اندکی اختصاصی‌تر است. این دستور بر روی کنترلری به نام TasksController، جهت ایجاد اکشن متدی به نام ArchiveTask با استفاده از کلاس ViewModel ایی به نام Task اجرا می‌شود.
حاصل آن ایجاد اکشن متد یاد شده به همراه کلاس TasksControllerTest است؛ البته اگر حین ایجاد پروژه جدید در ابتدای کار، گزینه ایجاد پروژه آزمون‌های واحد را نیز انتخاب کرده باشید. نام پروژه پیش فرضی که جستجوی می‌شود YourMvcProjectName.Test/Tests است.
نکته مهم آن، عدم حذف یا بازنویسی کامل کنترلر یاد شده است. کاری هم که در تولید متد آزمون واحد متناظر انجام می‌شود، تولید بدنه متد آزمون واحد به همراه تولید کدهای اولیه الگوی Arrange/Act/Assert است. پر کردن جزئیات بیشتر آن با برنامه نویس است.
و یا به صورت خلاصه‌تر:

```
PM> Scaffold UnitTest Tasks Delete
```

در اینجا متد آزمون واحد کنترلر Tasks و اکشن متد Delete آن، تولید می‌شود.

کار مقدماتی با MVC Scaffolding و امکانات مهیای در آن همینجا به پایان می‌رسد. در قسمت‌های بعد به سفارشی سازی این مجموعه خواهیم پرداخت.

نظرات خوانندگان

نویسنده: سهبلا صالح زاده
تاریخ: ۱۳۹۲/۰۶/۲۳ ۱۶:۰

در بخش EF Code First #11 عنوان کردید که مایکروسافت در تعریف DbContext اعلام می‌کند که DbSetها همان repository هستند و لایه ای دیگری ایجاد نشود، پس چرا در Scaffolding پارامتری برای آن در نظر گرفته است.

ببخشید من در استفاده از scaffolding در پروژه اصلی زمانی که کلاس‌ها را در پروژه دیگری تعریف می‌کنم مشکل دارم . خطا میدهد ولی اگر کلاس‌ها در یک پروژه تعریف شوند مشکلی ندارد.

نویسنده: سهبلا صالح زاده
تاریخ: ۱۳۹۲/۰۶/۲۳ ۱۶:۵

می‌خواستم بدونم در حالت One-to-many امکان استفاده از Html.EditForModel وجود دارد؟ یعنی میتوان بدون استفاده از UiHint ویا امثال اون فرم اتوماتیک ساخته شود و فیلدهای Dropdownlist را ایجاد کند چرا که در حالت عادی View به صورت EditForModel ساخته نشده و عناصر جدول وابسته به صورت لیست به View پاس داده می‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۲۳ ۱۶:۳۰

- لینک مطلب « پیاده سازی generic repository یک ضد الگو است » را براشون ارسال کنید تا مطالعه کنند.
- در متن عنوان شده « ModelType: برای ذکر صریح کلاس مورد استفاده در تشکیل کنترلر بکار می‌رود. اگر ذکر نشود، از نام کنترلر حدس زده خواهد شد.» ModelType دقیقا مانند نحوه مقدار دهی نوع مدل در صفحه دیالوگ استاندارد اضافه کردن یک View در VS.NET مقدار دهی می‌شود؛ یک fully qualified name است. با این شرط که اسمبلی مربوطه به پروژه اصلی ارجاع دارد و یکبار هم کل پروژه Build شده.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۲۳ ۱۶:۳۵

قسمت سوم این بحث به سفارشی سازی scaffolding پرداخته. اگر از پیش فرض‌های آن راضی نیستید یا هر تغییر خاصی را علاقمند بودید که به کلاس‌ها یا فایل‌های پیش فرض آن اعمال کنید، با سفارشی سازی قابل انجام است.

نویسنده: صالح زاده
تاریخ: ۱۳۹۲/۰۶/۲۴ ۱۶:۸

من خیلی سعی کردم اما نشد؛ مثلا کد زیر در پروژه DataLayer به درستی کار می‌کنه اما در پروژه اصلی با وجود Add شدن Refrence پروژه DataLayer کار نمی‌کنه و خطا میده.

مجبور میشم کدها را در DataLayer بسازم و بعد منتقل کنم به پروژه اصلی !

```
scaffold repository DataLayer.Models.City
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۲۴ ۱۸:۱۶

- سوئیچ ModelType رو ذکر نکردید. مثالش هست در متن (... Task *ModelType* - ...)

- خطاهایی رو هم که دریافت می‌کنید، اینجا به نویسنده اصلی گزارش بدید (به صورت کامل البته؛ نه اینکه صرفا عنوان کنید کار نمی‌کند).