

پیشنیاز:

[تعریف نوع جنریک به صورت متغیر](#)

مطلبی را چندی قبل در مورد نحوه خودکار کردن افزودن کلاس‌های EntityTypeConfiguration به modelBuilder در این سایت [مطالعه کردید](#) . در مطلب جاری به خودکار سازی تعاریف مرتبط با DbSet ها خواهیم پرداخت.

ابتدا مثال کامل زیر را درنظر بگیرید:

```
using System;
using System.Data.Entity;
using System.Data.Entity.Migrations;
using System.Linq;
using System.Reflection;

namespace MyNamespace
{
    public abstract class BaseEntity
    {
        public int Id { set; get; }
        public string CreatedBy { set; get; }
    }

    public class User : BaseEntity
    {
        public string Name { get; set; }
    }

    public class MyContext : DbContext
    {
        protected override void OnModelCreating(DbModelBuilder modelBuilder)
        {
            var asm = Assembly.GetExecutingAssembly();
            loadEntities(asm, modelBuilder, "MyNamespace");
        }

        void loadEntities(Assembly asm, DbModelBuilder modelBuilder, string nameSpace)
        {
            var entityTypees = asm.GetTypes()
                .Where(type => type.BaseType != null &&
                    type.Namespace == nameSpace &&
                    type.BaseType.IsAbstract &&
                    type.BaseType == typeof(BaseEntity))
                .ToList();

            var entityTypeMethod = typeof(DbModelBuilder).GetMethod("Entity");
            entityTypees.ForEach(type =>
            {
                entityTypeMethod.MakeGenericMethod(type).Invoke(modelBuilder, new object[] { });
            });
        }
    }

    public class Configuration : DbMigrationsConfiguration<MyContext>
    {
        public Configuration()
        {
            AutomaticMigrationsEnabled = true;
            AutomaticMigrationDataLossAllowed = true;
        }

        protected override void Seed(MyContext context)
        {
            context.Set<User>().Add(new User { Name = "name-1" });
            context.Set<User>().Add(new User { Name = "name-2" });
            context.Set<User>().Add(new User { Name = "name-3" });
            base.Seed(context);
        }
    }

    public static class Test
```

```
{
    public static void RunTests()
    {
        Database.SetInitializer(new MigrateDatabaseToLatestVersion<MyContext, Configuration>());
        using (var context = new MyContext())
        {
            var user1 = context.Set<User>().Find(1);
            if (user1 != null)
                Console.WriteLine(user1.Name);
        }
    }
}
```

توضیحات:

همانطور که ملاحظه می‌کنید در این مثال خبری از تعاریف DbSet ها نیست. به کمک Reflection تمام مدل‌های برنامه که از نوع کلاس پایه BaseEntity هستند (روشی مرسوم جهت مدیریت خواص تکراری مدل‌ها) یافت شده (در متد loadEntities) و سپس نتیجه حاصل به صورت پویا به متد جنریک Entity ارسال می‌شود. حاصل، افزوده شدن خودکار کلاس‌های مورد نظر به سیستم EF است.

البته در این حالت چون دیگر کلاس‌های مدل‌ها در MyContext به صورت صریح تعریف نمی‌شوند، نحوه استفاده از آن‌ها را توسط متد Set، در متدهای RunTests و یا Seed، ملاحظه می‌کنید.

نظرات خوانندگان

نویسنده:

محسن د.

تاریخ:

۲۳:۴۴ ۱۳۹۱/۰۸/۲۷

البته کد

```
var asm = Assembly.GetExecutingAssembly();
```

در صورتی کار میکند که مدلها در همین پروژه باشند ولی اگر در یک پروژه جداگانه تعریف شده باشند باید از

```
var asm = Assembly.GetAssembly(typeof(DomainModels.BaseEntity));
```

استفاده کرد . که DomainModels.BaseEntity یکی از کلاسهای موجود در اسمبلی مربوط به مدلها باید باشد .

نویسنده:

ali mazinani

تاریخ:

۱۴:۳۹ ۱۳۹۱/۰۸/۲۸

سلام آیا این کار سر بار اضافی نداره؟

نویسنده:

وحید نصیری

تاریخ:

۱۶:۱۴ ۱۳۹۱/۰۸/۲۸

نه. اینکار فقط یکبار در آغاز برنامه انجام میشود. نتیجه کار هم توسط EF کش خواهد شد.

نویسنده:

hoseini

تاریخ:

۲۳:۲۴ ۱۳۹۱/۱۲/۰۴

سلام

این متد کار افزودن نگاشتهای کلاسها رو هم انجام میدهد ؟

و اگر بخواد انجام بده باید چه تغییراتی روی کد انجام بشه؟

ممنون

نویسنده:

وحید نصیری

تاریخ:

۲۳:۳۶ ۱۳۹۱/۱۲/۰۴

« افزودن خودکار کلاسهای تنظیمات نگاشتهای در EF Code first »

نویسنده:

شایان مرادی

تاریخ:

۱۵:۵۱ ۱۳۹۱/۱۲/۲۵

سلام

اگر مدلها در چندین پروژه بودن چطور؟

مثلا به صورت مازول هر مازول مدل خود را دارد

نویسنده:

وحید نصیری

تاریخ: ۱۷:۱۷ ۱۳۹۱/۱۲/۲۵

(الف)

- تمام مدل‌های شما باید از کلاس مشخصی مثلا BaseEntity مشتق شوند.
- یا در تنظیمات برنامه مشخص کنید در حین Reflection چه فضای نامی باید جستجو شود.
- و یا مثلا مانند ASP.NET MVC اگر کلاسی نامش به عبارت خاصی ختم شد و همچنین از کلاس پایه خاصی نیز مشتق شده بود آنگاه بررسی شود.

(ب)

- مرحله بعد اندکی ویرایش متد loadEntities است جهت خواندن مدل‌های واقع شده مثلا در یک فضای نام خاص یا مشتق شده از یک کلاس پایه خاص و سپس افزودن خودکار آن‌ها به modelBuilder همانند چیزی که در مثال فوق پیاده سازی شده.
- در مثال بالا فقط یک اسمبلی جستجو می‌شود؛ اگر نیاز است تمام اسمبلی‌های بارگذاری شده در برنامه جستجو شوند، روش کار مراجعه به AppDomain است:

```
foreach (Assembly currentassembly in AppDomain.CurrentDomain.GetAssemblies())
{
    Type t = currentassembly.GetType("typeName", false, true);
    if (t != null) {return currentassembly.FullName;}
}
```

نویسنده: شایان مرادی

تاریخ: ۱۸:۱۱ ۱۳۹۱/۱۲/۲۵

ممنونم از جوابتون

AppDomain.CurrentDomain.GetAssemblies1 را از پوشه Bin اضافه می‌نماید یا اینکه جستجویی در کل Library ها انجام می‌دهد؟

نویسنده: وحید نصیری

تاریخ: ۱۸:۲۰ ۱۳۹۱/۱۲/۲۵

خیر. چیزی را اضافه یا بارگذاری نمی‌کند. فقط در AppDomain برنامه، کل اسمبلی‌ها و ماژول‌های بارگذاری شده موجود را [لیست می‌کند](#).

نویسنده: شایان مرادی

تاریخ: ۱۸:۲۳ ۱۳۹۱/۱۲/۲۶

سلام

من به این طریق Dbset ها رو اضافه میکنم و مشکلی پیش نمیاد . اما وقتی میخوام نگاشت‌ها را به وسیله توضیحات [این لینک](#) انجام بدم به مشکل میخورم
من ابتدا Dbset ها را با این کد شما اضافه میکنم سپس تابع اضافه کردن نگاشت‌ها را اجرا میکنم
اما به مشکل میخورم
ممنون میشم اگه راهنمایی نمایید به چه شکل باید هم موجودیت‌ها و هم نگاشت‌های آن‌ها را همزمان انجام داد
با تشکر

نویسنده: وحید نصیری

تاریخ: ۱۹:۰۱ ۱۳۹۱/۱۲/۲۶

عکس عمل کنید. ابتدا تنظیمات، بعد موجودیت‌ها اضافه شوند.
ضمنا اگر احیانا گذراتان به انجمنی افتاد این «به مشکل برخوردم» رو باید توضیح بدید. باید خطای حاصل رو ذکر کنید (علاوه بر روشی که طی شده).

نویسنده:

مهدی فرهانی

تاریخ:

۱۳:۵۹ ۱۳۹۲/۰۱/۰۹

با سلام

چندوقتی هست که دارم از این روش برای برنامه هام استفاده میکنم و هیچ مشکلی نداشتم تا اینکه دیشب با یک مشکل عجیب برخورد کردم و مشکل این بود که زمان ساخت جداول به Property که به صورت NotMapped در کلاس Base تعریف کرده بودم ایراد گرفت. پس از بررسی که انجام دادم متوجه شدم که ترتیب Map کردن Entity ها باعث این مشکل میشه.

<http://entityframework.codeplex.com/workitem/481>

این لینک مشکلی که گزارش شده، راه حلی که نوشته شده این که ترتیب Map کردن کلاس ها باید تغییر کند. این درحالی است که در این روش هیچ ترتیبی در نظر گرفته نمیشه و براساس خواندن کلاس ها از اسمبلی ساخته میشه. برای رفع این مشکل اومدم و یک ویژگی ترتیب به کلاس هایی که میدونستم ترتیبشون مهم هست اضافه کردم و زمان خواندن کلاس ها از اسمبلی مورد نظر اونها مرتب کردم و مشکل حل شد. آیا راه حل بهتری وجود داره یا نه ؟

```
var entityTypees = modelAssembly.GetTypes()
    .Where(type => type.Namespace != null
        && (type.Namespace.StartsWith(domainNamespace)
            && type.CustomAttributes.All(c => c.AttributeType !=
                typeof(ComplexTypeAttribute))
            && !type.IsAbstract && !type.IsEnum)
    )
    .OrderBy(c => c.CustomAttributes.Any(
        x => x.AttributeType == typeof (EntityOrderAttribute)) ?
        c.GetCustomAttribute<EntityOrderAttribute>().OrderNumber : 100).ToList();
```

نویسنده:

محسن

تاریخ:

۹:۹ ۱۳۹۲/۰۱/۱۰

این مساله در حالت معمولی تعریف DbSet ها هم پیش میاد. یعنی الزاما محدود به روش گفته شده در این مطلب نیست. خودشون هم پذیرفتن که یک باگ است و در حال رفعش هستند.

نویسنده:

پدرام جباری

تاریخ:

۲۱:۲۷ ۱۳۹۲/۰۲/۱۰

با سلام

من برای پروژه ای که در حال انجام اون هستم چون تعداد جداول زیاد هست برای آنکه زمان نمونه ساختن هر context کاهش پیدا کنه، برای هر بخش از پروژه Context متفاوتی ایجاد کردم تا زمان نگاشت تنظیمات جداول کاهش پیدا کنه ، البته یک Context کلی هست که اون وظیفه ساخت دیتابیس رو داره (داخل EF 6 این امکان هست که یک دیتابیس از چند Context متفاوت ایجاد شده باشه ، ولی خوب برای این کار تو EF5 باید یک Context کلی داشت که مشکلی ایجاد نشه) به نظر شما کار درستی هست ؟ اصلا تاثیری داره ؟

نویسنده:

وحید نصیری

تاریخ:

۲۱:۵۳ ۱۳۹۲/۰۲/۱۰

- بحث جاری در مورد EF 5 هست. کل مباحثی که در سایت مطرح شده در مورد [تا قبل از EF 6 است](#) . (هرچند هدف اصلی EF6 از بحث چند Context ایی پوشش دادن ایجاد جداول مختلف به ازای Schema های مختلف است. پیشتر dbo بیشتر مد نظر بود (پشتیبانی از تک schema به ازای یک دیتابیس). الان پوشش چندین Schema با هم در طی چندین Context مختلف در یک بانک اطلاعاتی)

- تاثیری نداره. نگاشت ها فقط یکبار در آغاز کار برنامه انجام می شوند و بعد کش خواهند شد. هر بار وهله سازی context به

معنای انجام چند باره نگاشت ها و یافتن و برقراری آنها نیست. اتفاقا این وهله سازی های ثانویه پس از آغاز برنامه، فوق العاده هم سریع هستند.

خلاصه اینکه مباحث مطرح شده در مطلب جاری فقط در آغاز برنامه اجرا می شوند و نه به ازای هر بار وهله سازی تک Context برنامه.

- در مورد اینکه چرا باید یک کلاس Context در برنامه داشت [اینجا توضیح دادم](#). بحث الگوی واحد کار مهم ترین آنها است.

نویسنده: پدرام جباری
تاریخ: ۱۳۹۲/۰۲/۱۰ ۲۳:۴۳

ممنون بابت جوابتون، واقعا نمی دونم چرا همیشه فکر می کردم به این صورت نیست و هر بار بخش نگاشت ها انجام میشه، تقریبا بیشتر مقاله های مربوط به افزایش سرعت رو در EF رو در وبسایت مطالعه کردم و کل روند انجام کار مفهوم شد برام، شاید بیشتر به خاطر عدم اطمینانم به EF بود که همچین فکری کردم.
واقعا ممنون بابت مطالبتون که در وبسایت قرار دادید خیلی به من کمک کرده از همه نظر.

نویسنده: بهنام حقی
تاریخ: ۱۳۹۳/۰۱/۲۸ ۱۲:۳۶

سلام

من یه مشکلی دارم توی این قسمت.

توی پروژه از خودکار سازی نگاشت ها و همین بخش استفاده کردم که در نهایت پیام خطای زیر رو میده:

The entity type User is not part of the model for the current context.

قسمت نگاشت ها همانند همین که تو سایت گفتین استفاده کردم. پروژه شامل سه بخش Domain و Model و Console هست برای تست این دو بخش. قسمت خودکار سازی نگاشت ها بدون خود کار سازی تعاریف DbSet ها به درستی کار میکنه، وقتی این بخش رو پیاده میکنم پیام خطای بالا رو میده تو متد Seed واسه دیتاهای پیش فرض User.

```
namespace Test.Domain
{
    public abstract class BaseEntity
    {
        public int Id { set; get; }
    }
}

namespace Test.Domain.Entities
{
    public class User : BaseEntity
    {
        public int UserNumber { get; set; }
        public string Name { get; set; }
    }
}

namespace Test.Domain.Mappings
{
    public class UserMap : EntityTypeConfiguration<User>
    {
        public UserMap()
        {
            this.HasKey(x => x.UserNumber);
            this.Property(x => x.Name).HasMaxLength(450).IsRequired();
        }
    }
}
```

```
loadEntities(asm, modelBuilder, "Test.Domain");
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۲۸ ۱۲:۵۱

درسته. چون کلاس User در فضای نام Test.Domain قرار ندارد.
بررسی انجام شده به صورت `type.Namespace == nameSpace` است ولی حالت مدنظر شما `type.Namespace.StartsWith` است.
`nameSpace` است.

نویسنده: بهنام حقی
تاریخ: ۱۳۹۳/۰۱/۲۸ ۱۳:۱۱

خیلی ممنون. برای اینکه همه اسمبلی ها رو بگرده، و به اسمبلی خاص رو بهش معرفی نکنیم، چطور از این تابع استفاده کنم:

```
foreach (Assembly asm in AppDomain.CurrentDomain.GetAssemblies())
{
    //...
}
```

تابع زیر رو به ازای هر فضای نام که میده باید فراخونی کنم؟

```
loadEntities(asm, modelBuilder, "نام فضا");
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۲۸ ۱۳:۲۲

بله. ولی این حالت مقرون به صرفه نیست. چون تعداد اسمبلی های بارگذاری شده در یک App domain زیاد است و شامل موارد پایه و سیستمی دات نت هم هست (با هزاران فضای نام). در این حالت این جستجو زمان زیادی را به خود اختصاص خواهد داد. بهتر است لیست یک سری اسمبلی مشخص را در نظر داشته باشید تا اینکه کل سیستم را جستجو کنید.

نویسنده: سدا
تاریخ: ۱۳۹۳/۰۳/۲۱ ۱۷:۱۵

اگر بخوایم خودکار DbSet ها تعریف شه و Models و Mapping در پروژه ای دیگه باشه، متود Seed که نیاز مستقیم به Context و DbSet ها داره. یعنی بر فرض بخوایم یک User به Context.Users اضافه کنیم امکان پذیر نیست درسته؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۲۱ ۱۷:۲۲

از متد Set استفاده کنید. در مثال فوق در متد `protected override void Seed` نمونه اش ارائه شده:

```
context.Set<User>().Add(new User { Name = "name-1" });
```