عنوان: **تنظیم رشته اتصالی Entity Framework به بانک اطلاعاتی به وسیله کد**
نویسنده: فلونی
تاریخ: ۱۳۹۳/۰۲/۲۶ ۱۴:۳۰
آدرس: www.dotnettips.info
گروه‌ها: Entity framework

در زمان ساخت مدل از بانک اطلاعاتی در روش Database First به صورت پیش فرض تنظیمات مربوط به اتصال (Connection String) مدل به بانک اطلاعاتی در فایل config برنامه ذخیره می‌شود. مشکل این روش آن است که در سیستم‌های مختلف، بسته به بستری که نرم افزار قرار است بر روی آن اجرا شود، باید تنظیمات مربوط به بانک اطلاعاتی صورت گیرد.
مثلا فرض کنید شما در زمان توسعه نرم افزار، SQL Server را به صورت Local بر روی سیستم خود نصب کرده اید و Connection String ساخته شده توسط ویزارد Entity Framework بر همین اساس ساخته و ذخیره شده‌است. حال بعد از انتشار برنامه، شخصی تصمیم دارد برنامه را بر روی سیستمی نصب کند که بانک اطلاعاتی Local نداشته و تصمیم به اتصال به یک بانک اطلاعاتی بر روی سرور دیگر یا با مشخصات (Login و Password و ...) دیگر را دارد. برای این مواقع نیاز به پیاده سازی روشی است تا کاربر نهایی بتواند تنظیمات مربوط به اتصال به بانک اطلاعاتی را تغییر دهد. روش‌های مختلفی مثل تغییر فایل app.config به صورت Runtime یا ... در سایت‌های مختلف ارائه شده که اکثرا روش‌های غیر اصولی و زمانبری جهت پیاده سازی هستند.
ساده‌ترین روش جهت انجام این کار، اعمال تغییری کوچک در Constructor کلاس مدل مشتق شده از DBContext می‌باشد. فرض کنید مدلی از بانک اطلاعاتی Personnely با نام PersonallyEntities ساخته اید که حاصل آن کلاس زیر خواهد بود:

```
public partial class PersonallyEntities : DbContext
    {
        public PersonallyEntities()
            : base("name=PersonallyEntities")
        {
        }
    }
```

همانطور که مشاهده می‌کنید، در Constructor این کلاس، نام Connection String مورد استفاده جهت اتصال به بانک اطلاعاتی به صورت زیر آورده شده که به Connection String ذخیره شده در فایل Config اشاره می‌کند:

```
"name=PersonallyEntities"
```

اگر به Connection String ذخیره شده در فایل Config دقت کنید متوجه می‌شوید که Connection String ذخیره شده، دارای فرمتی خاص و متفاوتی نسبت به Connection String معمولی ADO.NET است. متن ذخیره شده شامل تنظیمات و Metadata مدل ساخته شده جهت ارتباط با بانک اطلاعاتی نیز می‌باشد:

```
metadata=res://*/Model1.csdl|res://*/Model1.ssdl|res://*/Model1.msl;provider=System.Data.SqlClient;prov
ider connection string="data source=.;initial catalog=Personally;integrated
security=True;MultipleActiveResultSets=True;App=EntityFramework"
```

جهت تولید پویای Connection String، بسته به تنظیمات کاربر، نیاز است تا در آخر Connection String ی با فرمت بالا در اختیار Entity Framework قرار دهیم تا امکان اتصال به بانک فراهم شود. جهت تبدیل Connection String معمول ADO.NET به Connection String قابل فهم EF میتوان از کلاس EntityConnectionStringBuilder به صورت زیر استفاده کرد:

```
public static string BuildEntityConnection(string connectionString)
        {
            var entityConnection = new EntityConnectionStringBuilder
            {
                Provider = "System.Data.SqlClient",
                ProviderConnectionString = connectionString,
                Metadata = "res://*"
            };

            return entityConnection.ToString();
        }
```

همانطور که مشاهده می‌کنید، متد بالا با دریافت یک connectionString که همان ADO.NET ConnectionString ما می‌باشد،

تنظیمات و Metadata مورد نیاز Entity Framework را به آن اضافه کرده و یک EF ConnectionString برمی‌گرداند.
برای اینکه بتوان EF ConnectionString تولید شده را در هنگام اجرای برنامه به صورت Runtime اعمال کرد، نیاز است تا تغییر کوچکی در Constructor کلاس مدل تولید شده توسط Entity Framework ایجاد کرد. کلاس PersonnelyEntities به صورت زیر تغییر پیدا می‌کند:

```
public partial class PersonallyEntities : DbContext
    {
        public PersonallyEntities(string connectionString)
            : base(connectionString)
        {

        }
    }
```

با اضافه شدن پارامتر connectionString به سازنده کلاس PersonnelyEntities برای ساخت یک نمونه از مدل ساخته شده در کد نیاز است تا Connection String مورد نظر جهت برقراری ارتباط با بانک را به عنوان پارامتر، به متد سازنده پاس دهیم. سپس مقدار این پارامتر به کلاس والد (DbContext) جهت برقراری ارتباط با بانک اطلاعاتی ارجاع داده شده:

```
: base(connectionString)
```

در آخر به صورت زیر میتوان توسط EF به بانک اطلاعاتی مورد نظر متصل شد :

```
var entityConnectionString = BuildeEntityConnection("Data Source=localhost;Initial Catalog=Personally;
Integrated Security=True");
var PersonallyDb = new PersonallyEntities(entityConnectionString);
```

با این روش میتوان ADO Connection String مربوط به اتصال بانک اطلاعاتی را به راحتی به صورت داینامیک به وسیله اطلاعات وارد شده توسط کاربر و کلاس‌های تولید Connection String نظیر SQLConnectionStringBuilder تولید کرد و بدون تغییر در کدهای برنامه، به بانک‌های مختلفی متصل شد. همچنین با داینامیک کردن متد Provider کلاس EntityConnectionStringBuilder که در کد بالا با "System.Data.SqlClient" مقدار دهی شده، می‌توان وابستگی برنامه بانک اطلاعی خاص را از بین برد و بسته به تنظیمات مورد نظر کاربر، به موتورهای مختلف بانک اطلاعاتی متصل شد که البته لازمه این کار رعایت یکسری نکات فنی در پیاده سازی پروژه است که از حوصله این مقاله خارج است.
موفق باشید

نظرات خوانندگان

نویسنده: مهدی دهقانی
تاریخ: ۱۳۹۳/۰۲/۲۸ ۹:۲۸

سلام،
ممنون بابت تاپیک.
من یک پروژه web form دارم که به صورت 3 لایه پیاده سازی کردم و در لایه DAL از EF استفاده کردم (Database first). در لوکال مشکلی نیست. دیروز سایت رو آپلود کردم روی هاست، حالا نمیدونم که connectionString ای که هاست در اختیارم قرار داده رو به چه صورت جایگزین کنم.
امکانش هست راهنمایی کنید؟
یه سوال دیگه هم دارم، ازین روشی که گفتین اگه بخوام استفاده کنم، تکلیف connectionString ای که در AppConfig و Web.config (در قسمت UI) هستش چی میشه؟
ممنون

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۲/۲۸ ۱۳:۳۷

«به چه صورت جایگزین کنم»

مثل مثال بالا که از فایل کانفیگ گذاشته شد، قسمت provider connection string رو پیدا کن و جایگزینش کن با رشته اتصالی که به شما دادند.

«ازین روشی که گفتین اگه بخوام استفاده کنم»

این روش کاری به سناریوی شما نداره. برای جایی هست که برنامه قراره مثلا برای هر سال به یک دیتابیس مجزا وصل بشه. اول کار، کاربر باید انتخاب کنه که مثلا به دیتابیس سال 89 وصل بشه یا دیتابیس سال 92.

نویسنده: محمد حسابی
تاریخ: ۱۳۹۳/۰۲/۲۹ ۲۰:۲

من از پروژه سیستم مدیریت Iris که تو همین سایت ارائه شده استفاده می‌کنم و می‌خوام به 2 تا دیتابیس دسترسی داشته باشم. به خاطر همین 2 تا connectionString تو web.config ساختم ولی نمی‌دونم چطوری باید ازشون استفاده کنم که بتونم از هر دو دیتابیس استفاده کنم.

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۲/۲۹ ۲۱:۳۴

مطلب استفاده از چندین Context در EF 6 Code first شاید شروع خوبی باشه.

نویسنده: محمد حسابی
تاریخ: ۱۳۹۳/۰۲/۲۹ ۲۳:۲۲

ممنون بابت لینک، این قسمت رو از اون مقاله نقل قول می‌کنم: *«داشتن چندین Context در برنامه و مدیریت تراکنش‌ها در EF، هر DbContext معرف یک واحد کار است. یعنی تراکنش‌ها و چندین عمل متوالی مرتبط انجام شده، درون یک DbContext معنا پیدا می‌کنند. متد SaveChanges نیز بر همین اساس است که کلیه اعمال ردیابی شده در طی یک واحد کار را در طی یک تراکنش به بانک اطلاعاتی اعمال می‌کند.»*

این یعنی من باید 2 تا UnitOfWork داشته باشم؟ مثلاً IUnitOfWork1 و IUnitOfWork2 ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۲/۳۰ ۰:۳۱

به ازای هر دیتابیس با ساختار متفاوت باید یک DbContext جدا داشته باشی. بحث تزریق وابستگی‌ها جداست. می‌تونی named instance مثلا با استراکچرمپ درست کنی، بجای تعریف چندتا اینترفیس: For().Use().Named

نویسنده: سانای رحیمی
تاریخ: ۱۳۹۳/۰۵/۱۲ ۱۹:۴۷

سلام و سپاس بابت مطلب خوبتون
زمانی که از StructureMap و UoW استفاده می‌کنیم چگونه می‌توانیم پارامترهای Constructor را مقداردهی کنیم؟ من از روش زیر استفاده کردم ولی کار نکرد. ممنون میشم راهنمایی کنید.

```
ObjectFactory.Initialize(x =>
            {
                var ctx = new MyContext(GlobalVars.ConnectionString);
                x.For<IUnitOfWork>().Use(() => ctx);
x.For<IFactorForushMasterService>().Use<FactorForushMasterService>();
            });
            using (var container = ObjectFactory.Container.GetNestedContainer())
            {
                var uow = container.GetInstance<IUnitOfWork>();
                var factorService = container.GetInstance<IFactorForushMasterService>();
                txtShFactor.Text = factorService.GetLastShFactor().ToString();
                txtDateFactor.Text =
ShamsiDate.ConvertMiladiToShamsi(GeneralHelper.GetServerDateTime());
            }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۱۲ ۲۰:۳۶

```
// `connectionString` is the constructor's parameter name
cfg.For<IUnitOfWork>().Use<MyContext>().Ctor<string>("connectionString").Is(someValueAtRunTime);;
// or ...
ObjectFactory.Container.With("connectionString").EqualTo(someValueAtRunTime).GetInstance<IUnitOfWork>()
;
```

نویسنده: سانای رحیمی
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۲:۴۵

سلام
من به یک مشکلی خوردم. اون هم اینه که وقتی اولین بار برنامه رو اجرا می‌کنم که دیتابیس باید ساخته بشه، این اتفاق نمی‌افته. ولی وقتی رشته اتصال رو تنظیم نمی‌کنم دیتابیس ساخته میشه. ولی بعد از ساخت دیتابیس دیگه مشکلی پیش نمیاد و با روال عادی کار میکنه. میشه بگید دلیلش چی می‌تونه باشه؟
باز هم ممنون

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۳:۲۶

شاید مفید باشه : وادار کردن EF Code first به ساخت بانک اطلاعاتی پیش از شروع به کار برنامه