

در ادامه بحث ASP.NET MVC می‌شود به ابزاری به نام MVC Scaffolding اشاره کرد. کار این ابزار که توسط یکی از اعضای تیم ASP.NET MVC به نام [استیو اندرسون](#) تهیه شده، تولید کدهای اولیه یک برنامه کامل ASP.NET MVC از روی مدل‌های شما می‌باشد. حجم بالایی از کدهای تکراری آغازین برنامه را می‌شود توسط این ابزار تولید و بعد سفارشی کرد. MVC Scaffolding حتی قابلیت تولید کد بر اساس الگوی Repository و یا نوشتن Unit tests مرتبط را نیز دارد. بدیهی است این ابزار جای یک برنامه نویس را نمی‌تواند پر کند اما کدهای آغازین یک سری کارهای متداول و تکراری را به خوبی می‌تواند پیاده سازی و ارائه دهد. زیر ساخت این ابزار، علاوه بر ASP.NET MVC، آشنایی با Entity framework code first است.

در طی سری ASP.NET MVC که در این سایت تا به اینجا مطالعه کردید من به شدت سعی کردم از ابزارگرایی پرهیز کنم. چون شخصی که نمی‌داند مسیریابی چیست، اطلاعات چگونه به یک کنترلر منتقل یا به یک View ارسال می‌شوند، قراردادهای پیش فرض فریم ورک چیست یا زیر ساخت امنیتی یا فیلترهای ASP.NET MVC کدامند، چطور می‌تواند از ابزار پیشرفته Code generator ایی استفاده کند، یا حتی در ادامه کدهای تولیدی آن‌را سفارشی سازی کند؟ بنابراین برای استفاده از این ابزار و درک کدهای تولیدی آن، نیاز به یک پیشنهاد دیگر هم وجود دارد: «Entity framework code first»

امسال دو کتاب خوب در این زمینه منتشر شده‌اند به نام‌های:

[Programming Entity Framework: DbContext](#) , ISBN: 978-1-449-31296-1

[Programming Entity Framework: Code First](#) , ISBN: 978-1-449-31294-7

که هر دو به صورت اختصاصی به مقوله EF Code first پرداخته‌اند.

در طی روزهای بعدی EF Code first را با هم مرور خواهیم کرد و البته این مرور مستقل است از نوع فناوری میزبان آن؛ می‌خواهد یک برنامه کنسول باشد یا WPF یا یک سرویس ویندوز NT و یا ... یک برنامه وب. البته از دیدگاه میکروسافت، M در MVC به معنای EF Code first است. به همین جهت MVC3 به صورت پیش فرض ارجاعی را به اسمبلی‌های آن دارد و یا حتی به روز رسانی که برای آن ارائه داده نیز در جهت تکمیل همین بحث است.

مروری سریع بر تاریخچه Entity framework code first

ویژوال استودیو 2010 و دات نت 4، به همراه EF 4.0 ارائه شدند. با این نگارش امکان استفاده از حالت‌های طراحی database first و model first مهیا است. پس از آن، به روز رسانی‌های EF خارج از نوبت و به صورت منظم، هر از چندگاهی ارائه می‌شوند و در زمان نگارش این مطلب، آخرین نگارش پایدار در دسترس آن 4.3.1 می‌باشد. از زمان EF 4.1 به بعد، نوع جدیدی از مدل سازی به نام Code first به این فریم ورک اضافه شد و در نگارش‌های بعدی آن، مباحث DB migration جهت ساده سازی تطابق اطلاعات مدل‌ها با بانک اطلاعاتی، اضافه گردیدند. در روش Code first، کار با طراحی کلاس‌ها که در اینجا مدل داده‌ها نامیده می‌شوند، شروع گردیده و سپس بر اساس این اطلاعات، تولید یک بانک اطلاعاتی جدید و یا استفاده از نمونه‌ای موجود میسر می‌گردد.

پیشتر در روش database first ابتدا یک بانک اطلاعاتی موجود، مهندسی معکوس می‌شد و از روی آن فایل XML ایی با پسوند EDMX تولید می‌گشت. سپس به کمک entity data model designer ویژوال استودیو، این فایل نمایش داده شده و یا امکان اعمال تغییرات بر روی آن میسر می‌شد. همچنین در روش دیگری به نام model first نیز کار از entity data model designer جهت طراحی موجودیت‌ها آغاز می‌گشت.

اما با روش Code first دیگر در ابتدای امر مدل فیزیکی و یک بانک اطلاعاتی وجود خارجی ندارد. در اینجا EF تعاریف کلاس‌های شما را بررسی کرده و بر اساس آن، اطلاعات نگاشت‌های خواص کلاس‌ها به جداول و فیلدهای بانک اطلاعاتی را تشکیل می‌دهد. البته عموماً تعاریف ساده کلاس‌ها بر این منظور کافی نیستند. به همین جهت از یک سری متادیتا به نام ویژگی‌ها یا اصطلاحاً data annotations مهیا در فضای نام System.ComponentModel.DataAnnotations برای افزودن اطلاعات لازم مانند نام فیلدها، جداول و یا تعاریف روابط ویژه نیز استفاده می‌گردد. به علاوه در روش Code first یک API جدید به نام Fluent API نیز جهت تعاریف این

ویژگی‌ها و روابط، با کدنویسی مستقیم نیز در نظر گرفته شده است. نهایتاً از این اطلاعات جهت نگاشت کلاس‌ها به بانک اطلاعاتی و یا برای تولید ساختار یک بانک اطلاعاتی خالی جدید نیز می‌توان کمک گرفت.

مزایای EF Code first

- مطلوب برنامه نویسی‌ها! : برنامه نویسی‌هایی که مدتی تجربه کار با ابزارهای طراح را داشته باشند به خوبی می‌دانند این نوع ابزارها عموماً demo-ware هستند. چنداناً کلیک می‌کنید، دوبار Next، سه بار OK و ... به نظر می‌رسد کار تمام شده. اما واقعیت این است که عمری را باید صرف نگهداری و یا پیاده سازی جزئیاتی کرد که انجام آن‌ها با کدنویسی مستقیم بسیار سریعتر، ساده‌تر و با کنترل بیشتری قابل انجام است.
- سرعت: برای کار با EF Code first نیازی نیست در ابتدای کار بانک اطلاعاتی خاصی وجود داشته باشد. کلاس‌های خود را طراحی و شروع به کدنویسی کنید.
- سادگی: در اینجا دیگر از فایل‌های EDMX خبری نیست و نیازی نیست مرتباً آن‌ها را به روز کرده یا نگهداری کرد. تمام کارها را با کدنویسی و کنترل بیشتری می‌توان انجام داد. به علاوه کنترل کاملی بر روی کد نهایی تهیه شده نیز وجود دارد و توسط ابزارهای تولید کد، ایجاد نمی‌شوند.
- طراحی بهتر بانک اطلاعاتی نهایی: اگر طرح دقیقی از مدل‌های برنامه داشته باشیم، می‌توان آن‌ها را به المان‌های کوچک و مشخصی، تقسیم و refactor کرد. همین مساله در نهایت مباحث database normalization را به نحوی مطلوب و با سرعت بیشتری میسر می‌کند.
- امکان استفاده مجدد از طراحی کلاس‌های انجام شده در سایر ORM‌های دیگر. چون طراحی مدل‌های برنامه به بانک اطلاعاتی خاصی گره نمی‌خورند و همچنین الزاماً هم قرار نیست جزئیات کاری EF در آن‌ها لحاظ شود، این کلاس‌ها در صورت نیاز در سایر پروژه‌ها نیز به سادگی قابل استفاده هستند.
- ردیابی ساده‌تر تغییرات: روش اصولی کار با پروژه‌های نرم افزاری همواره شامل استفاده از یک ابزار سورس کنترل مانند SVN، Git، مرکوریال و امثال آن است. به این ترتیب ردیابی تغییرات انجام شده به سادگی توسط این ابزارها میسر می‌شوند.
- ساده‌تر شدن طراحی‌های پیچیده‌تر: برای مثال پیاده سازی ارث بری، ایجاد کلاس‌های خود ارجاع دهنده و امثال آن با کدنویسی ساده‌تر است.

دریافت آخرین نگارش EF

برای دریافت و نصب آخرین نگارش EF نیاز است از [NuGet](#) استفاده شود و این مزایا را به همراه دارد: به کمک NuGet امکان با خبر شدن از به روز رسانی جدید صورت گرفته به صورت خودکار در نظر گرفته شده است و همچنین کار دریافت بسته‌های مرتبط و به روز رسانی ارجاعات نیز در این حالت خودکار است. به علاوه توسط NuGet امکان دسترسی به کتابخانه‌هایی که مثلاً در گوگل کد قرار دارند و به صورت معمول امکان دریافت آن‌ها برای ما میسر نیست، نیز بدون مشکل فراهم است (برای نمونه ELMAH، که اصل آن از گوگل کد قابل دریافت است؛ اما بسته نیوگت آن نیز در دسترس می‌باشد).

پس از نصب NuGet، تنها کافی است بر روی گره References در Solution explorer ویژوال استودیو، کلیک راست کرده و به کمک NuGet آخرین نگارش EF را نصب کرد. در گالری آنلاین آن، عموماً EF اولین گزینه است (به علت تعداد بالای دریافت آن).

حین استفاده از NuGet جهت نصب Ef، ابتدا ارجاعاتی به اسمبلی‌های زیر به برنامه اضافه خواهند شد:

System.ComponentModel.DataAnnotations.dll

System.Data.Entity.dll

EntityFramework.dll

بدیهی است بدون استفاده از NuGet، تمام این کارها را باید دستی انجام داد.

سپس در پوشه‌ای به نام packages، فایل‌های مرتبط با EF قرار خواهند گرفت که شامل اسمبلی آن به همراه ابزارهای DB Migration است. همچنین فایل packages.config که شامل تعاریف اسمبلی‌های نصب شده است به پروژه اضافه می‌شود. NuGet به کمک این فایل و شماره نگارش درج شده در آن، شما را از به روز رسانی‌های بعدی مطلع خواهد ساخت:

```
<?xml version="1.0" encoding="utf-8"?>
<packages>
  <package id="EntityFramework" version="4.3.1" />
</packages>
```

همچنین اگر به فایل app.config یا web.config برنامه نیز مراجعه کنید، یک سری تنظیمات ابتدایی اتصال به بانک اطلاعاتی در آن ذکر شده است:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <!-- For more information on Entity Framework configuration, visit
http://go.microsoft.com/fwlink/?LinkID=237468 -->
    <section name="entityFramework"
type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=4.3.1.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" />
  </configSections>
  <entityFramework>
    <defaultConnectionFactory type="System.Data.Entity.Infrastructure.SqlConnectionFactory,
EntityFramework">
      <parameters>
        <parameter value="Data Source=(localdb)\v11.0; Integrated Security=True;
MultipleActiveResultSets=True" />
      </parameters>
    </defaultConnectionFactory>
  </entityFramework>
</configuration>
```

همانطور که ملاحظه می‌کنید بانک اطلاعاتی پیش فرضی که در اینجا ذکر شده است، [LocalDB](#) می‌باشد. این بانک اطلاعاتی را از [این آدرس](#) نیز می‌توانید دریافت کنید.

البته ضرورتی هم به استفاده از آن نیست و از سایر نگارش‌های SQL Server نیز می‌توان استفاده کرد ولی خوب ... مزیت استفاده از آن برای کاربر نهایی این است که «نیازی به یک مهندس برای نصب، راه اندازی و نگهداری ندارد». تنها مشکل آن این است که از ویندوز XP پشتیبانی نمی‌کند. البته SQL Server CE 4.0 این محدودیت را ندارد. ضمن اینکه باید در نظر داشت EF به فناوری میزبان خاصی گره نخورده است و مثال‌هایی که در اینجا بررسی می‌شوند صرفاً تعدادی برنامه کنسول معمولی هستند و نکات عنوان شده در آن‌ها در تمام فناوری‌های میزبان موجود به یک نحو کاربرد دارند.

قراردادهای پیش فرض EF Code first

عنوان شد که اطلاعات کلاس‌های ساده تشکیل دهنده مدل‌های برنامه، برای تعریف جداول و فیلدهای یک بانک اطلاعات و همچنین مشخص سازی روابط بین آن‌ها کافی نیستند و مرسوم است برای پر کردن این خلاء از یک سری متادیتا و یا Fluent API مهیا نیز استفاده گردد. اما در EF Code first یک سری قرار داد توکار نیز وجود دارند که مطلع بودن از آن‌ها سبب خواهد شد تا حجم کدنویسی و تنظیمات جانبی این فریم ورک به حداقل برسند. برای نمونه مدل‌های معروف بلاگ و مطالب آن‌را در نظر بگیرید:

```
namespace EF_Sample01.Models
{
    public class Post
    {
        public int Id { set; get; }
        public string Title { set; get; }
        public string Content { set; get; }
        public virtual Blog Blog { set; get; }
    }
}
```

```
using System.Collections.Generic;

namespace EF_Sample01.Models
{
```

```
public class Blog
{
    public int Id { set; get; }
    public string Title { set; get; }
    public string AuthorName { set; get; }
    public IList<Post> Posts { set; get; }
}
}
```

یکی از قراردادهای EF Code first این است که کلاس‌های مدل شما را جهت یافتن خاصیتی به نام Id یا ClassId مانند BlogId، جستجو می‌کند و از آن به عنوان primary key و فیلد identity جدول بانک اطلاعاتی استفاده خواهد کرد. همچنین در کلاس Blog، خاصیت لیستی از Posts و در کلاس Post خاصیت virtual ایی به نام Blog وجود دارند. به این ترتیب روابط بین دو کلاس و ایجاد کلید خارجی متناظر با آن‌را به صورت خودکار انجام خواهد داد. نهایتاً از این اطلاعات جهت تشکیل database schema یا ساختار بانک اطلاعاتی استفاده می‌گردد. اگر به فضاهای نام دو کلاس فوق دقت کرده باشید، به کلمه Models ختم شده‌اند. به این معنا که در پوشه‌ای به همین نام در پروژه جاری قرار دارند. یا مرسوم است کلاس‌های مدل را در یک پروژه class library مجزا به نام DomainClasses نیز قرار دهند. این پروژه نیازی به ارجاعات اسمبلی‌های EF ندارد و تنها به اسمبلی System.ComponentModel.DataAnnotations.dll نیاز خواهد داشت.

EF Code first چگونه کلاس‌های مورد نظر را انتخاب می‌کند؟

ممکن است ده‌ها و صدها کلاس در یک پروژه وجود داشته باشند. EF Code first چگونه از بین این کلاس‌ها تشخیص خواهد داد که باید از کدامیک استفاده کند؟ اینجا است که مفهوم جدیدی به نام DbContext معرفی شده است. برای تعریف آن یک کلاس دیگر را به پروژه برای مثال به نام Context اضافه کنید. همچنین مرسوم است که این کلاس را در پروژه class library دیگری به نام DataLayer اضافه می‌کنند. این پروژه نیاز به ارجاعی به اسمبلی‌های EF خواهد داشت. در ادامه کلاس جدید اضافه شده باید از کلاس DbContext مشتق شود:

```
using System.Data.Entity;
using EF_Sample01.Models;

namespace EF_Sample01
{
    public class Context : DbContext
    {
        public DbSet<Blog> Blogs { set; get; }
        public DbSet<Post> Posts { set; get; }
    }
}
```

سپس در اینجا به کمک نوع جنریکی به نام DbSet، کلاس‌های دومین برنامه را معرفی می‌کنیم. به این ترتیب، EF Code first ابتدا به دنبال کلاسی مشتق شده از DbContext خواهد گشت. پس از یافتن آن، خواصی از نوع DbSet را بررسی کرده و نوع‌های متناظر با آن‌را به عنوان کلاس‌های دومین در نظر می‌گیرد و از سایر کلاس‌های برنامه صرف‌نظر خواهد کرد. این کل کاری است که باید انجام شود.

اگر دقت کرده باشید، نام کلاس‌های موجودیت‌ها، مفرد هستند و نام خواص تعریف شده به کمک DbSet، جمع می‌باشند که نهایتاً متناظر خواهند بود با نام جداول بانک اطلاعاتی تشکیل شده.

تشکیل خودکار بانک اطلاعاتی و افزودن اطلاعات به جداول

تا اینجا بدون تهیه یک بانک اطلاعاتی نیز می‌توان از کلاس Context تهیه شده استفاده کرد و کار کدنویسی را آغاز نمود. بدیهی

است جهت اجرای نهایی کدها، نیاز به یک بانک اطلاعاتی خواهد بود. اگر تنظیمات پیش فرض فایل کانفیگ برنامه را تغییر ندهیم، از همان defaultConnectionFactory پاده شده استفاده خواهد کرد. در این حالت نام بانک اطلاعاتی به صورت خودکار تنظیم شده و مساوی «EF_Sample01.Context» خواهد بود.

برای سفارشی سازی آن نیاز است فایل app.config یا web.config برنامه را اندکی ویرایش نمود:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    ...
  </configSections>
  <connectionStrings>
    <clear/>
    <add name="Context"
          connectionString="Data Source=(local);Initial Catalog=testdb2012;Integrated Security = true"
          providerName="System.Data.SqlClient"
        />
  </connectionStrings>
  ...
</configuration>
```

در اینجا به بانک اطلاعاتی testdb2012 در وهله پیش فرض SQL Server نصب شده، اشاره شده است. فقط باید دقت داشت که تگ configSections باید در ابتدای فایل قرار گیرد و مابقی تنظیمات پس از آن.

یا اگر علاقمند باشید که از SQL Server CE استفاده کنید، تنظیمات رشته اتصالی را به نحو زیر مقدار دهی نمایید:

```
<connectionStrings>
  <add name="MyContextName"
        connectionString="Data Source=|DataDirectory|\Store.sdf"
        providerName="System.Data.SqlServerCe.4.0" />
</connectionStrings>
```

در هر دو حالت، name باید به نام کلاس مشتق شده از DbContext اشاره کند که در مثال جاری همان Context است.

یا اگر علاقمند بودید که این قرارداد توکار را تغییر داده و نام رشته اتصالی را با کدنویسی تعیین کنید، می‌توان به نحو زیر عمل کرد:

```
public class Context : DbContext
{
    public Context()
        : base("ConnectionStringName")
    {
    }
}
```

البته ضرورتی ندارد این بانک اطلاعاتی از پیش موجود باشد. در اولین بار اجرای کدهای زیر، به صورت خودکار بانک اطلاعاتی و جداول Blogs و Posts و روابط بین آنها تشکیل می‌گردد:

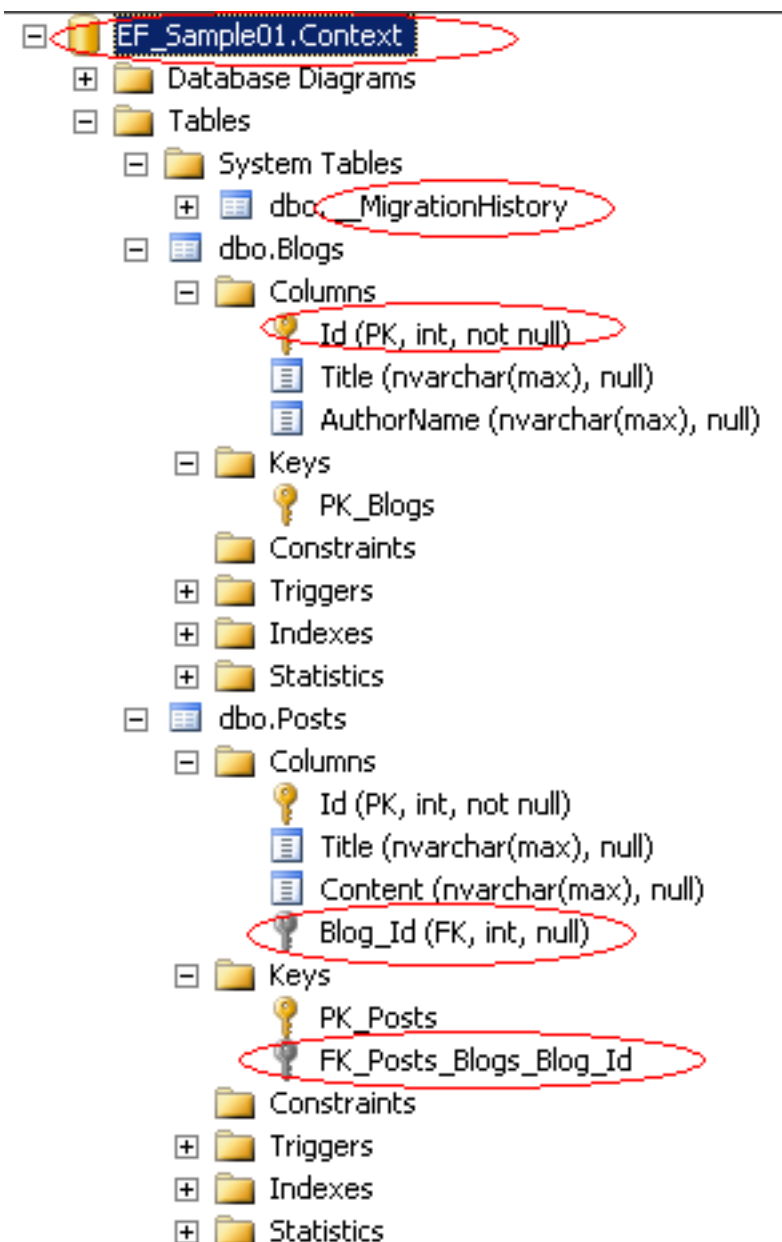
```
using EF_Sample01.Models;

namespace EF_Sample01
{
    class Program
    {
        static void Main(string[] args)
        {
        }
```

```

using (var db = new Context())
{
    db.Blogs.Add(new Blog { AuthorName = "Vahid", Title = ".NET Tips" });
    db.SaveChanges();
}
}
}

```



در این تصویر چند نکته حائز اهمیت هستند:

الف) نام پیش فرض بانک اطلاعاتی که به آن اشاره شد (اگر تنظیمات رشته اتصالی قید نگردد).

ب) تشکیل خودکار primary key از روی خواصی به نام Id

ج) تشکیل خودکار روابط بین جداول و ایجاد کلید خارجی (به کمک خاصیت virtual تعریف شده)

د) تشکیل جدول سیستمی به نام 'dbo.__MigrationHistory' که از آن برای نگهداری سابقه به روز رسانی‌های ساختار جداول کمک گرفته خواهد شد.

ه) نوع و طول فیلدهای متنی، 'nvarchar' از نوع 'max' است.

تمام این‌ها بر اساس پیش فرض‌ها و قراردادهای توکار EF Code first انجام شده است.

در کدهای تعریف شده نیز، ابتدا یک وهله از شیء Context ایجاد شده و سپس به کمک آن می‌توان به جدول Blogs اطلاعاتی را افزود و در آخر ذخیره نمود. استفاده از using هم در اینجا نباید فراموش شود، زیرا اگر استثنایی در این بین رخ دهد، کار پاکسازی منابع و بستن اتصال گشوده شده به بانک اطلاعاتی به صورت خودکار انجام خواهد شد. در ادامه اگر بخواهیم مطلبی را به Blog ثبت شده اضافه کنیم، خواهیم داشت:

```
using EF_Sample01.Models;

namespace EF_Sample01
{
    class Program
    {
        static void Main(string[] args)
        {
            //addBlog();
            addPost();
        }

        private static void addPost()
        {
            using (var db = new Context())
            {
                var blog = db.Blogs.Find(1);
                db.Posts.Add(new Post
                {
                    Blog = blog,
                    Content = "data",
                    Title = "EF"
                });
                db.SaveChanges();
            }
        }

        private static void addBlog()
        {
            using (var db = new Context())
            {
                db.Blogs.Add(new Blog { AuthorName = "Vahid", Title = ".NET Tips" });
                db.SaveChanges();
            }
        }
    }
}
```

متد db.Blogs.Find، بر اساس primary key بلاگ ثبت شده، یک وهله از آن را یافته و سپس از آن جهت تشکیل شیء Post و افزودن آن به جدول Posts استفاده می‌شود. متد Find ابتدا Contxet جاری را جهت یافتن شیء‌ای با id مساوی یک جستجو می‌کند (اصطلاحاً به آن first level cache هم گفته می‌شود). اگر موفق به یافتن آن شد، بدون صدور کوئری اضافی به بانک اطلاعاتی از اطلاعات همان شیء استفاده خواهد کرد. در غیراینصورت نیاز خواهد داشت تا ابتدا کوئری لازم را به بانک اطلاعاتی ارسال کرده و اطلاعات شیء Blog متناظر با id=1 را دریافت کند. همچنین اگر نیاز داشتیم تا تنها با سطح اول کش کار کنیم، در EF Code first می‌توان از خاصیتی به نام Local نیز استفاده کرد. برای مثال خاصیت db.Blogs.Local بیانگر اطلاعات موجود در سطح اول کش می‌باشد.

نهایتاً کوئری Insert تولید شده توسط آن به شکل زیر خواهد بود (لاگ شده توسط برنامه [SQL Server Profiler](#)):

```
exec sp_executesql N'insert [dbo].[Posts]([Title], [Content], [Blog_Id])
values (@0, @1, @2)
select [Id]
from [dbo].[Posts]
where @@ROWCOUNT > 0 and [Id] = scope_identity()',
N'@@ nvarchar(max) ,@1 nvarchar(max) ,@2 int',
@0=N'EF',
```

```
@1=N'data',
@2=1
```

این نوع کوئری‌های پارامتری چندین مزیت مهم را به همراه دارند:

الف) به صورت خودکار تشکیل می‌شوند. تمام کوئری‌های پشت صحنه EF پارامتری هستند و نیازی نیست مرتباً مزایای این امر را گوشزد کرد و باز هم عده‌ای با جمع زدن رشته‌ها نسبت به نوشتن کوئری‌های نا امن SQL اقدام کنند.

ب) کوئری‌های پارامتری در مقابل حملات تزریق اس کیوال مقاوم هستند.

ج) SQL Server با کوئری‌های پارامتری همانند رویه‌های ذخیره شده رفتار می‌کند. یعنی query execution plan محاسبه شده آن‌ها را کش خواهد کرد. همین امر سبب بالا رفتن کارایی برنامه در فراخوانی‌های بعدی می‌گردد. الگوی کلی مشخص است. فقط پارامترهای آن تغییر می‌کنند.

د) مصرف حافظه SQL Server کاهش می‌یابد. چون SQL Server مجبور نیست به ازای هر کوئری اصطلاحاً Ad Hoc رسیده یکبار execution plan متفاوت آن‌ها را محاسبه و سپس کش کند. این مورد مشکل مهم تمام برنامه‌هایی است که از کوئری‌های پارامتری استفاده نمی‌کنند؛ تا حدی که گاهی تصور می‌کنند شاید SQL Server دچار نشستی حافظه شده، اما مشکل جای دیگری است.

مشکل! ساختار بانک اطلاعاتی تشکیل شده مطلوب کار ما نیست.

تا همینجا با حداقل کدنویسی و تنظیمات مرتبط با آن، پیشرفت خوبی داشته‌ایم؛ اما نتیجه حاصل آنچنان مطلوب نیست و نیاز به سفارشی سازی دارد. برای مثال طول فیلدها را نیاز داریم به مقدار دیگری تنظیم کنیم، تعدادی از فیلدها باید به صورت not null تعریف شوند یا نام پیش فرض بانک اطلاعاتی باید مشخص گردد و مواردی از این دست. با این موارد در قسمت‌های بعدی بیشتر آشنا خواهیم شد.

نظرات خوانندگان

نویسنده: مهمان
تاریخ: ۱۳۹۱/۰۲/۱۴ ۱۰:۵۴:۲۱

با سلام و خسته نباشید. امید است این سری مطالب هم مانند مطالب MVC فراتر از مقالات و کتب موجود باشد.
سه سوال:

- 1- چطور می توان با Code First برخی از موارد ابتدایی ایجاد بانک مانند Collation و Compatibility Level و Schema و User و Role را به DBMS ارسال کرد.
- 2- اگر از پروایدهای دیگر مثلا برای MySQL یا Oracle استفاده شود، آیا Code First قادر است بدون هیچ تغییری نسبت به SQL Server کد را به بانکهای دیگر نگاشت کند؟ در مورد بانک های NOSQL چطور؟
- 3- آیا اگر این پروژه Code First را در یک هاست اشتراکی Deploy کنیم و در آن هاست برنامه Start شود (مثلا یک پروژه MVC)، آیا پروژه قادر خواهد بود به طور خودکار بانک را تولید نماید و دیگر نخواهیم بصورت دستی بانک و یوزر را در کنترل پنل هاست تعریف کنیم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۲/۱۴ ۱۱:۵۹:۴۸

- 1 و 3 - در انتهای بحث عرض کردم در قسمت های بعدی خیلی از موارد رو توضیح خواهم داد. این قسمت اول و فقط یک «مقدمه» ابتدایی بود.
- 2 - EF با بانک های اطلاعاتی NoSQL کار نمی کند. ضمنا هستند بانک های اطلاعاتی NoSQL ایی که برای دات نت نوشته شده اند و از همان روز اول با کلاس ها و LINQ کار می کنید مانند RavenDB. طراحی فوق العاده ای داره (^) . استفاده از EF Code first با سایر بانک های اطلاعاتی بجز مشتقات SQL Server نیز میسر است. برای آن ها نیاز به پروایدر مخصوص وجود دارد؛ مثلا: (^)

نویسنده: محمدی
تاریخ: ۱۳۹۱/۰۲/۱۴ ۱۲:۲۳:۲۳

تغییرات در کدها (دیتابیس) چگونه مدیریت می شوند(بروزرسانی)؟ یکی از کارهای سخت بروز رسانی دیتابیس مشتری امیدوارم EF راه مناسبی برای این موضوع داشته باشه.
مقادیر پیش فرض در دیتابیس کی و چگونه مدیریت می شوند؟
این سوالات تو ذهنم پیش اومد اینجا نوشتم که در قسمت های بعد جوابشون رو بگیرم .
مطئنم مثل همیشه چیزهای زیادی اینجا یاد میگیرم ، مرسی

نویسنده: MehdiPayervand
تاریخ: ۱۳۹۱/۰۲/۱۴ ۱۲:۴۱:۳۱

با عرض سلام و خسته نباشید، جناب مهندس با وجود اینکه معرفیتون در سطح خیلی بالایی هست ولی جای خالی مقایسه با NHibernate رقیب کهنه کار و اپن سورس EF خالیه، باز هم ممنون

نویسنده: علی قمشلویی
تاریخ: ۱۳۹۱/۰۲/۱۴ ۱۴:۵۹:۵۵

با سلام و تشکر در مورد استفاده از POCO نیز لطف کنید مطلب بزارید

نویسنده: Iman Amirdarabi
تاریخ: ۱۳۹۱/۰۲/۱۴ ۱۶:۲۲:۵۲

با سلام

ممنون از مطالب مفید شما.
 من کلا با اصل code first مشکل دارم.
 صرف وقت کم درسته برنامه سریع تر بالا می یاد ولی از طرف دیگه باید وقت بیشتری صرف توسعه مواردی کرد که توسط نویسنده فریم ورک پیش بینی نشده.
 مشکل دومی که وجود داره در code first خیلی از ویژگی هایی که در هر دیتابیس وجو داره از دست داده می شه مثل استفاده از sp
 در آخر یک سوال آیا شما حاضری یک برنامه پیچیده تحت وب با این مدل پیاده سازی کنی به عنوان یک مهندس صاحب نظر؟

نویسنده: وحید نصیری
 تاریخ: ۱۶:۳۶:۴۵ ۱۳۹۱/۰۲/۱۴

- یکی از اهداف ORM ها این است که برنامه رو مستقل از بانک اطلاعاتی پیاده سازی کنند؛ تا بتوان در صورت نیاز راحت به یک بانک اطلاعاتی دیگر سوئیچ کرد. من اینجا وقت نگذاشتم در مورد «چرا باید از ORM استفاده کرد» توضیح بدم مانند (^) یا مانند (^) و ... باز هم جستجو کنید هست.
 - بله. عدم استفاده از یک ORM در پروژه این روزها اشتباه محض است.

نویسنده: Naser Tahery
 تاریخ: ۱۹:۳۸:۱۴ ۱۳۹۱/۰۲/۱۴

بی صبرانه منتظر قسمت های بعدی هستیم.
 ممنون از شما

نویسنده: ZB
 تاریخ: ۲۳:۲۷:۳۴ ۱۳۹۱/۰۲/۱۴

سلام آقای نصیری
 ممنون بخاطر مطالب مفیدتون. EF یک ORM قوی هست ولی بعضی مواردش هست که به نظر بنده نوعی در حد این ORM نیست. که دو موردش رو در اینجا ذکر میکنم :
 (1) در برنامه های چند لایه اگر لایه ها بصورت پروژه های جداگانه در نظر گرفته شده باشند باید Connection String رو در لایه UI ، DAL و قرار داد که به نظر من از لحاظ امنیتی درست نیست. این بهتره که در UI ما اصلا ندونیم Connection String چی هست. البته این مورد با کد نویسی قابل حل هست ولی در کل مناسب نیست
 (2) من در کی از پروژه های گروهی به این مسئله برخوردم. ما در قسمت DAL فلدرهای مختلف داشتیم و هر کسی در فلدر قسمتی که کار میکرد میخواست یک دیتا مدل داشته باشه. این کار باعث میشد که به تعداد دیتا مدلها ما Connection String داشته باشیم و اگر میخواستیم از یک Connection String استفاده کنیم اسم کلاسی که تولید میشد برای همه یکسان میبود (اما در Name Space های مختلف) .

به نظر من EF از لحاظ Connection String یک مقدار ضعف داره. تا نظر دوستان چه باشه. ممنون و موفق باشید

نویسنده: وحید نصیری
 تاریخ: ۲۳:۴۹:۴۵ ۱۳۹۱/۰۲/۱۴

Connection String کاری به دیتامدل نداره. پیش فرض آن نام کلاسی است که از DbContext مشتق می شود (در مطلب فوق توضیح دادم: «در هر دو حالت، name باید به نام کلاس مشتق شده از DbContext اشاره کند که در مثال جاری همان Context است.»).
 نیازی هم نیست در سراسر پروژه تکرار شود. یکبار باید در فایل کانفیگ برنامه تعریف شود.
 اطلاع داشتن از این قراردادهای توکار از اتلاف وقت جلوگیری می کند.

نویسنده: وحید نصیری

تاریخ: ۱۳۹۱/۰۲/۱۴ ۲۳:۵۱:۵۳

ضمن اینکه زمانیکه از ORM استفاده می‌کند لایه DAL همان ORM است و نیازی نیست کار اضافه‌تری انجام دهید. این لایه خودبخود لحاظ شده است.

نویسنده: رضا
تاریخ: ۱۳۹۱/۰۲/۱۵ ۱۰:۳۲:۲۰

بسیار عالی. من در مورد code first در خود سایت asp.net خوندم. اما توضیحات شما بخصوص چون به زبان فارسی است خیلی در درک اونچه درست نفهمیده بودم کمک کرد. منتظر بخش های بعدی هستیم.

نویسنده: Salehi
تاریخ: ۱۳۹۱/۰۲/۱۵ ۱۱:۵۳:۱۹

برای اینکه بانک اطلاعاتی به طور خودکار تشکیل بشه، اجرای هر دستوری که به بانک مربوط میشه کافیه؟ مثلا دستور select ؟ بعد از اون به ازای هربار اجرای یک دستور، وجود یا عدم وجود DB چک میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۲/۱۵ ۱۲:۰۳:۳۲

در قسمت دوم این سری تحت عنوان «استراتژی‌های مقدماتی تشکیل بانک اطلاعاتی در EF Code first» توضیح دادم.

نویسنده: Iman Amirdarabi
تاریخ: ۱۳۹۱/۰۲/۱۵ ۱۴:۳۸:۳۲

سلام استاد
وقتی قرار برنامه مستقل از دیتابیس باشه ویژگی های هر دیتابیس چی می شه
مثل sp ها ufn ها یا type هایی که مختص یک دیتابیس و

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۲/۱۵ ۱۶:۱۶:۵۷

- این ویژگی‌ها رو می‌تونید فراموش کنید. چون مثلا SP در SQL Server CE وجود خارجی ندارد. اما برنامه‌ی نوشته شده با EF به راحتی می‌تونه با انواع و اقسام بانک‌های اطلاعاتی که پروایدر EF برای آن‌ها مهیا باشد، کار کند. برنامه شما به دیتابیس خاصی گره نمی‌خوره. اگر لازم بود راحت می‌تونید با تغییر پروایدر و تغییر کانکشن استرینگ، بدون نیازی به تغییر در کدهای خود، از یک بانک اطلاعاتی دیگر استفاده کنید.

- در EF Code first امکان استفاده از SP و امثال آن هم وجود دارد (در جای خودش توضیح خواهم داده به چه نحوی). البته در این حالت برنامه فقط مختص به SQL Server خواهد شد.

نویسنده: ZB
تاریخ: ۱۳۹۱/۰۲/۱۶ ۱۴:۰۰:۴۳

با تشکر از پاسختون اما باید عرض کنم همونطور که مطلع هستید Connection String در EF مثل Linq 2 SQL نیست که به یک رشته خلاصه باشه بلکه مسیرهای CSDL، SSDL، MSL را هم لازم دارد. بنابراین اگر چند دیتا مدل داشته باشیم مجبوریم که چند Connection String ذخیره کنیم. دومین مطلبی که باید عرض کنم اینه که شما براحتی به مشکل خورد برنامه در فقدان Connection String رو در لایه های بالاتر میتوانید تست کنید. البته شما استاد هستید برای دوستان تازه کار عرض میکنم که در یک Solution دو پروژه اضافه کنید یکی برای دیتا مدل و یکی هم برای واسط کاربر. چنانچه از پروژه واسط بخواهید توابعی رو از دیتا مدل صدا بزنید به شما ارور برخواهد گشت و شما باید Connection String رو به برنامه واسط اضافه کنید. با تشکر

نویسنده: وحید نصیری

تاریخ: ۱۴:۲۲:۴۱ ۱۳۹۱/۰۲/۱۶

بحث من در اینجا EF Code first است. در اینجا شما دیگر با SSDL و غیره کاری ندارید. در مقدمه عرض کردم روش‌های database first و model first هم بودند و هستند. این فرق می‌کند. code first هست. بحث چیز دیگری است.

نویسنده: آرش
تاریخ: ۲۲:۲۲:۳۶ ۱۳۹۱/۰۲/۱۶

با درود و سپاس از شما - اگر مقدور بود لطفاً یک توضیح کوچک در مورد LocalDB و تفاوت اون با sql ce بفرمایید.

نویسنده: وحید نصیری
تاریخ: ۰۰:۰۱:۴۹ ۱۳۹۱/۰۲/۱۷

یک جدول مقایسه‌ای اینجا هست در این مورد: ([^](#))

نویسنده: Sirwan Afifi
تاریخ: ۰۰:۳۰:۳۰ ۱۳۹۱/۰۲/۲۲

سلام استاد خیلی ممنون بابت آموزشهاتون
یه سوال :

همونطور که توضیح دادید در کل ما سه نوع پروژه لازم داریم : 1- Domain Classes که حاوی Model های ما هست 2- DataLayer که حاوی کلاس Context می باشد و در نهایت پروژه خودمان

حال مشکل من اینجاست که در داخل کلاس Context که ایجاد کرده ام کلاس DbContext و رفرنس EF_Sample01.Models (نام پروژه رو همون EF_Sample01 گذاشتم یعنی داخل یک Solution این سه نوع پروژه رو دارم) رو نمی شناسه.

نویسنده: وحید نصیری
تاریخ: ۰۰:۳۷:۴۶ ۱۳۹۱/۰۲/۲۲

DbContext نیاز به ارجاعی به اسمبلی EF دارد که باید به این class library های دیگر هم اضافه شود.

نویسنده: مشعل
تاریخ: ۱۱:۲۰ ۱۳۹۱/۰۴/۰۴

با سلام
تشکیل خودکار بانک اطلاعاتی و جداول برای من انجام نمی‌شود. در واقع چون دیتابیس مورد نظر که در Connection String نام برده شده وجود ندارد، برنامه من اصلاً به دیتابیس کانکت نمی‌شود و با خطای زیر هنگام اجرای متد SaveChanges مواجه می‌شوم:
Cannot open database "EFTest" requested by the login. The login failed.

لطفاً راهنمایی بفرمائید
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۱:۵۹ ۱۳۹۱/۰۴/۰۴

در مورد کانکشن استرینگ، ایجاد بانک اطلاعاتی و غیره در قسمت‌های بعدی بیشتر توضیح داده شده. اگر تعاریف رشته اتصالی شما به این نحو باشد:

<connectionStrings>

```
<clear/>
<add
  name="ProductContext"
  connectionString="Data Source=(local);Initial Catalog=testdb2012;Integrated Security = true"
  providerName="System.Data.SqlClient"
/>
</connectionStrings>
```

به این معنا است که کلاس Context شما به نحو زیر تعریف شده است:

```
public class ProductContext : DbContext
```

بنابراین جزو قرار دادهای توکار EF Code first است که name ذکر شده در قسمت تعریف رشته اتصالی در فایل کانفیگ، با نام کلاس مشتق شده از DbContext یکی باشد.

با این تعاریف باید برنامه کار کند (البته بر اساس نام کلاسهای برنامه شما).

ضمناً login failed به این معنا است که رشته اتصالی اشتباه تعریف شده است. رشته فوق به یک بانک اطلاعاتی sql server و به وهله پیش فرض آن اشاره می‌کند و از نوع windows authentication است. این موارد را باید بر اساس تنظیمات سیستم خودتون تغییر بدید.

نویسنده: torisoft
تاریخ: ۲۳:۵۹ ۱۳۹۱/۰۴/۱۰

سلام جناب نصیری

سیلور 5 از code first پشتیبانی نمی‌کند ؟ موقع نصب از nuget پیغام می‌ده که

Could not install package 'EntityFramework 4.3.1'. You are trying to install this package into a project that targets 'Silverlight,Version=v5.0', but the package does not contain any assembly references that are compatible with that framework.

نویسنده: وحید نصیری
تاریخ: ۰:۷ ۱۳۹۱/۰۴/۱۱

سیلورلایت به صورت مستقیم با هیچ نوع ORM ایی کار نمی‌کند (چون یک فناوری سمت کاربر است). اما شما در سمت سرور می‌تونید به کمک یک WCF سرویس و مشتقات مشابه آن با EF یا NH و غیره کار کنید و سپس نتیجه را در یک برنامه سیلورلایت مصرف کنید.

نویسنده: mina
تاریخ: ۱۳:۱۴ ۱۳۹۱/۰۴/۲۸

سلام ممنون از مطالب مفیدتون

من تازه 1 روزه شروع کردم code first کار کنم

تو به مقاله داشتم می‌خوندم برای تعریف entity ها این association ها رو تو هر دو طرف virtual تعریف کرده بود

مثلا برای blog این طور نوشته بود

```
public virtual IList<post> posts { set; get; }
```

خودم هم این طور نوشتم تا این جا که فرقی ندیدم می‌شه توضیح بدید چه فرقی دارن

باز هم ممنونم

نویسنده: وحید نصیری
تاریخ: ۱۳:۳۷ ۱۳۹۱/۰۴/۲۸

مراجعه کنید به [قسمت 10](#) این سری، بحث lazy loading آن.

نویسنده: هوشنگ
تاریخ: ۱۲:۵۹ ۱۳۹۱/۰۵/۰۱

سلام ،

جناب نصیری آیا من میتونم از EF Code First با دیتابیس Oracle استفاده کنم یا خیر ؟

نویسنده: وحید نصیری
تاریخ: ۱۳:۳۲ ۱۳۹۱/۰۵/۰۱

بله. نیاز به [پروایدر خود شرکت اوراکل](#) را دارد و کار می‌کند.

نویسنده: فرید صالحی
تاریخ: ۱۳:۲۶ ۱۳۹۱/۰۵/۱۶

با سلام. از بین دو کتابی که در ابتدا معرفی کردید، من code first رو مطالعه کردم. با توجه به اینکه قصد دارم دوره شما رو هم کامل مطالعه کنم، نیاز به مطالعه کتاب DbContext هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳:۵۵ ۱۳۹۱/۰۵/۱۶

هرچقدر بیشتر مطالعه کنید بهتر است.

نویسنده: امیر هاشم زاده
تاریخ: ۰:۳۴ ۱۳۹۱/۰۵/۱۷

چه پارامترهایی در انتخاب Code First، Model First، DataBase First برای یک پروژه وجود دارد؟

نویسنده: شاهین کیاست
تاریخ: ۲:۵۶ ۱۳۹۱/۰۵/۱۷

اگر قبلا نخواندید ، خواندن این [سوال و جواب](#) خالی از لطف نیست.

چه دلیل قانع کننده ای وجود داره برای پروژه ای که از صفر شروع می‌شود از روشی غیر از Code First استفاده شود ؟ حتی برای پروژه هایی که پایگاه داده‌ی آنها وجود دارد هم ابزارهای مهندسی معکوس (جهت تولید خودکار موجودیت ها) وجود دارد.

نویسنده: امیر هاشم زاده
تاریخ: ۱:۱۲ ۱۳۹۱/۰۵/۱۸

اشاره‌ی خوبی بود، ولی بیشتر به محک می‌خواستم که امکان تشخیص این وجود داشته باشه که برای چه پروژه‌ای از چه رهیافتی استفاده کنیم. البته مطالب سوال و جواب هم تا حدی کمک میکنه

سلام مهندس نصیری، چرا این کد توی EF5 خطای کلید خارجی میده؟
کدش از کتاب Code First که معرفی کردین استفاده کردم اما کد خودتون خطا نداره

```
using System;
using System.Collections.Generic;
namespace ChapterOneProject
{
    public class Patient
    {
        public Patient()
        {
            Visits = new List<Visit>();
        }

        public int Id { get; set; }
        public string Name { get; set; }
        public DateTime BirthDate { get; set; }
        //[ForeignKey("AnimalTypeId")]

        public AnimalType AnimalType { get; set; }
        //public int AnimalTypeId { get; set; }

        public DateTime FirstVisit { get; set; }
        public List<Visit> Visits { get; set; }
    }

    public class Visit
    {
        [Key]
        public int Id { get; set; }
        public DateTime Date { get; set; }
        public String ReasonForVisit { get; set; }
        public String Outcome { get; set; }
        public Decimal Weight { get; set; }

        //[ForeignKey("PatientId")]
        //public virtual Patient Patient { get; set; }
        public int PatientId { get; set; }
    }

    public class AnimalType
    {
        public int Id { get; set; }
        public string TypeName { get; set; }
    }
}
```

کد کانتکست

```
public class VetContext : DbContext
{
    public DbSet<Patient> Patients { get; set; }
    public DbSet<Visit> Visits { get; set; }
    //public DbSet<AnimalType> AnimalTypes { get; set; }
}
```

و در تابع Main برنامه Console این نوشته شده اما خطا میده و ثبت نمی‌شه

```
var dog = new AnimalType { TypeName = "Dog" };
var visit = new List<Visit>
{
    new Visit
    {
        Date = new DateTime(2011, 9, 1),
        Outcome = "Test",
        ReasonForVisit = "Test",
        Weight = 32,
```

```

    }
};
var patient = new Patient
{
    Name = "Sampson",
    BirthDate = new DateTime(2008, 1, 28),
    AnimalType = dog,
    Visits = visit,
};
using (var context = new VetContext())
{
    context.Patients.Add(patient);
    context.SaveChanges();
}

```

کدهای دیگه تست کردم مشکلی نداشت اما این مورد ؟
با profiler چک کردم خطای عدم توانایی در تبدیل نوع datetime2 به datetime میده

نویسنده: صابر فتح الهی
تاریخ: ۱۵:۴۷ ۱۳۹۱/۱۲/۰۷

اشکالاش پیدا کردم توی کتاب برای شی patient چون خصوصیت FirstVisit مقداردهی نشده و نباید تهی باشد بنابراین زمان اجرا نمی‌تواند تاریخ پیش فرض را تبدیل کند. با اضافه کردن خط زیر

```

....
BirthDate = new DateTime(2008, 1, 28),
<<< FirstVisit = new DateTime(2008,1,12), >>>
AnimalType = dog,
.....

```

مشکل حل شد.

نویسنده: محسن
تاریخ: ۱۶:۳۸ ۱۳۹۱/۱۲/۰۷

DateTime در دات نت value type هست یعنی نال پذیر نیست مگر اینکه Nullable تعریف شود.

نویسنده: امیر
تاریخ: ۱۳:۲۲ ۱۳۹۱/۱۲/۲۳

چطوری من جداول رو به دیتابیس اضافه کنم
کلاس رو نوشتم و وب کانفیگ رو هم ست کردم مثل ef اسکریپت نداره چطوری ادد کنم

نویسنده: وحید نصیری
تاریخ: ۱۳:۳۳ ۱۳۹۱/۱۲/۲۳

- جداول رو خودش اضافه می‌کنه به صورت خودکار؛ در اولین بار اجرای برنامه.
- برای سفارشی سازی یا تهیه اسکریپت، در قسمت‌های 4 و 5 این سری به تفصیل بحث شده.

نویسنده: امیر
تاریخ: ۱۴:۵۷ ۱۳۹۱/۱۲/۲۳

زمان اجرا این خطا رو میده
 .The type initializer for 'System.Data.Entity.Internal.AppConfig' threw an exception
 مشکل از کجاست

نویسنده: وحید نصیری
 تاریخ: ۱۶:۴۸ ۱۳۹۱/۱۲/۲۳

- حداقل دو علت می‌تونه داشته باشه:
 الف) از پروژه‌ای استفاده می‌کنید که از چند ماژول تشکیل شده. اولی به EF نگارش A ارجاع دارد دومی به EF نگارش B. همه این‌ها رو باید یک دست کنید.
 ب) EF رو به روز کردید اما تعریف آن‌را در فایل کانفیگ به روز نکردید. برای مثال این تعریف قدیمی در فایل کانفیگ شما هست

```
<section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=4.3.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"/>
```

که در آن به EF 4.3 اشاره شده. بعد پروژه رو به EF 5 آپدیت کردید اما این مورد به روز نشده که باید به صورت زیر تغییر کند:

```
<section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=5.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
```

ج) تعریف ConnectionStrings در فایل کانفیگ باید بعد از ConfigSections باشد و نه قبل از آن.

- ضمناً تمام مثال‌های این سری [از اینجا](#) قابل دریافت است.

نویسنده: امیر
 تاریخ: ۱۸:۲۶ ۱۳۹۱/۱۲/۲۳

اقای نصیری مرسی که واقعا وقت میزاری. واقعا زکات علم تو میدی.

sectionname="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection,>
 app.config در بالا <"/EntityFramework, Version=4.3.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
 استفاده میشه که مال ویندوزه من asp.net کار میکنم پس اضافه بود پاکش کردم درست شد
 بازم تشکر میکنم

نویسنده: امیر
 تاریخ: ۱۳:۵ ۱۳۹۲/۰۱/۰۷

با سلام
 من میخام پی دی اف EF Code frist رو تا آخرین درس داشته باشم چیکار باید کرد لینکشو بی زحمت میزارین مثل Asp.net mvc

نویسنده: وحید نصیری
 تاریخ: ۱۳:۰۹ ۱۳۹۲/۰۱/۰۷

لطفاً مراجعه کنید به [ابتدای گروه EF](#) در سایت. لینک دریافت PDF کلیه مطالب گروه به صورت یکجا قرار دارد. این نکته در مورد سایر گروه‌های سایت هم صادق است.

نویسنده: محمد
 تاریخ: ۱۲:۳۷ ۱۳۹۲/۰۱/۱۹

سلام

وقتی کانکشن استرینگو به این صورت تعریف می‌کنم :

```
<configuration>
<configSections>
</configSections>
<connectionStrings>
<clear/>
<add name="Context" connectionString="Data Source=localhost;Initial Catalog=test;Integrated Security =
true" providerName="System.Data.SqlClient"/>
</connectionStrings>
<system.web>
<compilation debug="true"/></system.web>
</configuration>
```

این error می‌دهد :

An error occurred while getting provider information from the database. This can be caused by Entity Framework using an incorrect connection string. Check the inner exceptions for details and ensure that the connection string is correct.
علتش چی می‌تونه باشه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۱۹ ۱۲:۵۷

خودش گفته چکار کنی. باید به inner exceptions مراجعه کنید. این تازه سطح اول خطا است. این خطا تو در تو است و تمام خطاهای EF تو در تو طراحی شدن. مابقی رو هم در یک انجمن پیگیری کنید. نیاز هست کدت باشه. نیاز هست مشخص باشه سطح دسترسی‌ات به کانکشن استرینگ که تعریف کردی برقرار است یا نه و خیلی از مسایل دیگه.

نویسنده: debugger
تاریخ: ۱۳۹۲/۰۱/۲۰ ۱۳:۱۹

سلام
با توجه به خطایی که گذاشته شده به نظر مشکل از ConnectionString هست و اگر مثال این قسمت رو انجام دادی و instance شما به غیر از (local) است هنگام نوشتن نام DataSource بایستی پرانتزها رو برداری

```
<add name="Context"
connectionString="Data Source=.\sql2012;Initial Catalog=testdb2012;Integrated Security = true"
providerName="System.Data.SqlClient"
/>
```

موفق باشید

نویسنده: م.ر
تاریخ: ۱۳۹۲/۰۲/۰۲ ۰:۵

سلام. اصلاً یاد گرفتن code first خوب هست یا نه؟ آیا پیاده سازی‌های اون تو پروژه مشکلی ایجاد نمیکنه؟ در مقابل روش db first چه مزیتها و معایبی داره؟ بهترین منبع یادگیری برای ef کجاست؟ راستی مفهوم change tracking تو ef رو میتونید توضیح بدید؟
متشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۲ ۰:۳۵

- از اون بهتر نیست .
- قدم به قدم. عجله نکن. [در قسمت 14](#) این سری بهش پرداخته شده.

نویسنده: م.ر.
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۰:۵۱

سلام.
آقا ممنون از جواب .
ببینید من مطلب لینک رو خوندم . پرداخته به انتخاب orm.
اما حالا من صحبت‌م متمرکز هست روی خود ef. من منظورم اینه که کجا باید codeFirst استفاده بشه کجا dbFirst. آیا کار کردن روی یک پروژه بصورت codeFirst در آینده یعنی وقتی حجم دیتا و ارتباطات زیاد شد مشکلی نخواهد ساخت؟ سرعت کدام بهتر است؟ آیا با وجود قابلیت‌های lambda , linq نیازی به ساخت storedProcedure سمت دستابیس اصلا داریم یا نه؟
متشکرم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۱:۱۵

[یک موتور اصلی EF](#) بیشتر وجود نداره. برای کار کردن با این موتور اصلی سه روش حداقل تدارک دیده شده:
(الف) database first: مربوط به زمانی که یک دیتابیس از پیش طراحی شده با ساختار جداول و ارتباطات آن موجود است. این روش، ساختار بانک اطلاعاتی شما رو مهندسی معکوس کرده و یک سری کد و دیاگرام را برای استفاده توسط موتور اصلی EF تولید می‌کنه.
(ب) روش model first: در VS.NET می‌تونید طراحی‌های مرتبط با جداول، کلاس‌ها و روابط رو انجام بدید. کدهای اضافی برای کار با موتور EF و همچنین به روز رسانی بانک اطلاعاتی رو به صورت خودکار انجام خواهد داد.
(ج) روش code first: در این روش دیگر خبری از طراح بصری نیست. کار با کدنویسی و طراحی کلاس‌ها و ایجاد روابط بین آن‌ها توسط برنامه نویس شروع می‌شود. نهایتاً این کلاس‌ها توسط موتور EF استفاده خواهند شد. امکان تبدیل این کلاس‌ها به بانک اطلاعاتی متناظر و همچنین به روز رسانی خودکار بانک اطلاعاتی با تغییر ساختار کلاس‌ها هم پیش بینی شده. روش code first بهترین حالت است برای کسانی که نمی‌خواهند از انبوهی از کدهای تولید شده به صورت خودکار (حاصل از مهندسی معکوس یک بانک اطلاعاتی موجود) استفاده کنند و می‌خواهند کنترل بیشتری بر روابط و اختصاصی سازی آن‌ها داشته باشند. در این حالت می‌تونید بدون نیاز به یک بانک اطلاعاتی یک برنامه را کامل کنید (منهای مباحث تست سیستم).
روش code first در حال حاضر روشی است که بیشتر توسط تیم EF تبلیغ می‌شود و در حال توسعه است. مابقی رو هم کم کم دارند تبدیل می‌کنند به پسته‌ای برای حالت code first. مثلاً ابزار تهیه کردند برای مهندسی معکوس یک بانک اطلاعاتی موجود به روش code first. کد نهایی تمیزتری داره؛ چون کلاس‌ها را خودتان طراحی می‌کنید و توسط ابزارها به صورت خودکار تولید نمی‌شوند، کنترل بیشتر و نهایتاً کیفیت بالاتری دارند. ساده است؛ درگیر نگهداری edmx modelها نخواهید بود. به روز رسانی بانک اطلاعاتی آن هم می‌تواند کاملاً خودکار شود.

برای اطلاعات بیشتر در مورد مزایای این روش یا تاریخچه EF متن قسمت جاری را یکبار مطالعه کرده و قسمت‌های «مروری سریع بر تاریخچه Entity framework code first» و «مزایای EF Code first» را بررسی کنید.
+ کل قسمت EF [از اینجا](#) به صورت یک فایل PDF قابل دریافت است. در مورد اکثر مواردی که عنوان کردید به صورت مجزا بحث شده و توضیحات کافی ارائه شدن.

نویسنده: محبوبه محمدی
تاریخ: ۱۳۹۲/۰۲/۱۰ ۱۶:۸

ممنون از مطلب خوبتون.
برای جدول‌های زیاد که توی یک DbContext اضافه شدند، برای اولین لود و ذخیره زمان خیلی زیادی گرفته میشه و البته pre-generating views هم فایده نداشته! و از طرف دیگه توی برنامه مجبورم برای اینکه دیتا رو از دیتابیس بگیرم یکبار

Reload (DbEntityEntry<TEntity>(entityToRefresh).Reload()) کنم. من نمی‌فهمم یعنی همه‌ی دیتا یکبار توی شروع برنامه از دیتابیس دریافت میشن که دفعه‌های بعدی از Cash خونده میشن؟! ممنون میشم اگر یکم در این مورد توضیح بدید و جایی رو معرفی کنید که بشه مقدار بیشتر در مورد first level cache خوند.

نویسنده: وحید نصیری
تاریخ: ۱۶:۲۹ ۱۳۹۲/۰۲/۱۰

[سایت جاری](#) از EF Code first استفاده می‌کنه. مشکلی هم با کارایی آن وجود ندارد. برای مسایل شخصی نیاز به بررسی کدهای شما، بررسی best practices، بررسی‌های ویژه توسط EF Profilers و همچنین code review هست. به عبارتی نیاز به مشاور خصوصی دارید. موفق باشید

نویسنده: ahmadb7
تاریخ: ۸:۲۰ ۱۳۹۲/۰۲/۳۱

با سلام کدام را شما ترجیح می‌دهید EF یا NH و برای شروع کدام بهتر است با تشکر

نویسنده: وحید نصیری
تاریخ: ۹:۰۳ ۱۳۹۲/۰۲/۳۱

[توضیحات در اینجا](#)

نویسنده: احمد احمدی
تاریخ: ۲۳:۳ ۱۳۹۲/۰۴/۰۶

من هم به همین مشکل خوردم. یکی از دلایلش نال بودن مقدار DateTime مدل هست. [DbUpdateException : MVC/EF Code First](#) [SaveChanges](#) :

نویسنده: وحید نصیری
تاریخ: ۰:۳۰ ۱۳۹۲/۰۴/۰۷

در EF تا Inner exception را بررسی نکنید، دلیل اصلی را مشاهده نخواهید کرد و نهایتاً با سعی و خطا و حدس و گمان، پیش خواهید رفت.

نویسنده: محمدیوسف میرجلیلی
تاریخ: ۱۴:۵۷ ۱۳۹۲/۰۵/۱۵

سلام. در درس ششم کلاس‌های کانفیگ را در فضای نام Mappings تعریف کردیم. اگر پروژه شامل چند اسمبلی باشه (DomainClasses و DataLayer)، فضای نام Mappings و کلاس‌های مرتبط با اون را بهتره در کدوم یک از پروژه‌ها ایجاد کنیم؟

نویسنده: وحید نصیری
تاریخ: ۱۵:۱۱ ۱۳۹۲/۰۵/۱۵

در قسمت 12 این سری با مثال قابل دریافت توضیح داده شده.

نویسنده: امیر خلیلی
تاریخ: ۱۱:۲۱ ۱۳۹۲/۰۸/۱۹

یک سوال ابتدایی

آیا سرعت کار با دیتابیس و فراخوانی دیتا با استفاده از EF نسبت به ADO.Net و یا همان DataSet و DataReader بیشتر است ؟
یا فقط به خاطر یک سری مزیت‌های دیگه باید رو بیاریم به این تکنولوژی ؟
راستش من فقط سرعت کار برام مهمه !

نویسنده: وحید نصیری
تاریخ: ۱۱:۳۱ ۱۳۹۲/۰۸/۱۹

تمام ORM‌های دات نت در سطح پایین خودشون از ADO.NET استفاده می‌کنند. بنابراین پشت صحنه این‌ها، استفاده از Data Reader و امثال آن است، به همراه یک سری مزیت جانبی دیگه مانند: « [5 دلیل برای استفاده از یک ابزار ORM](#) » و جلوگیری از اختراع چرخ‌هایی به شدت ناقص و معیوب مانند: « [مروری بر کدهای کلاس SqlHelper](#) » و همچنین امنیت توکار و پیش فرض لحاظ شده در آن‌ها مانند: « [امنیت در LINQ to SQL](#) »

نویسنده: محمد رضا خزائی
تاریخ: ۲۱:۱۸ ۱۳۹۲/۱۰/۱۳

با سلام؛ منم این مشکل رو دارم. برای بار اول دیتابیس رو نمیسازه.

نویسنده: وحید نصیری
تاریخ: ۲۱:۲۵ ۱۳۹۲/۱۰/۱۳

« [وادر کردن EF Code first به ساخت بانک اطلاعاتی پیش از شروع به کار برنامه](#) »

نویسنده: XPlan
تاریخ: ۱۱:۵۵ ۱۳۹۲/۱۱/۱۱

سلام

1-می خواستم بدونم برای مثال در کلاس Blog شما

```
public class Blog
{
    public int Id { set; get; }
    public string Title { set; get; }
    public string AuthorName { set; get; }
    public IList<Post> Posts { set; get; }
}
```

EF دقیقاً چه زمانی (و با فراخوانی چه متد هایی) از اکسسورهای set و چه زمانی از get استفاده می‌کند؟
2- اگر در همین کلاس Blog به هر دلیل نیاز باشد که از اکسسورهای خودکار #C استفاده نکنیم کلاس Blog چگونه خواهد شد؟ لطفاً این کلاس را بدون اکسسورهای خودکار باز نویسی کنید

```
get
{
    return ?
}
set
{
    //push calculated private field to db ?
}
```

نویسنده: وحید نصیری
تاریخ: ۱۲:۲۵ ۱۳۹۲/۱۱/۱۱

- با استفاده از امکانات [Reflection](#) دات نت، در زمان خواندن اطلاعات از بانک اطلاعاتی، از set و زمان دریافت اطلاعات از کاربر

و تشکیل کوئری SQL نهایی از get استفاده خواهد کرد.

- در قسمت سوم این سری، در مورد فیلدهای محاسباتی بحث شده « [DatabaseGenerated و NotMapped \(6\)](#) »

نویسنده: منصور جعفری
تاریخ: ۱۳۹۲/۱۱/۱۱ ۱۲:۵۲

سلام! آیا روش Code First برای ویندوز اپلیکشن هم استفاده میشه...؟
ممنونم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۱ ۱۳:۱۸

در هر فناوری مرتبط با دات نت که کل دات نت فریم ورک در دسترس باشد، قابل استفاده است. از WPF تا WinForms تا WCF و انواع و اقسام برنامه‌های وب؛ از ویندوز سرویس تا یک برنامه‌ی کنسول ساده.

نویسنده: ح مراداف
تاریخ: ۱۳۹۲/۱۱/۲۱ ۱۶:۵

سلام،
منظور شما اینه که باید با Nuget رفرنس Entity Framework رو روی هر سه پروژه (Domain Classes و DataLayer و پروژه اصلی) نصب کنم ؟

من وب اپلیکیشن تازه داره کار می‌کنم و تا الان همش وب سایت کار می‌کردم، آیا بصورت پیش فرض EntityFramework توی پروژه‌ها وجود نداره و حتما باید با Nuget رفرنس اونو به پروژه اضافه کنیم ؟
(یعنی این dll با نصب ویژوال استودیو نصب نمیشه؟! و باید از نوگت دانلودش کنیم؟)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۲۱ ۱۶:۴۱

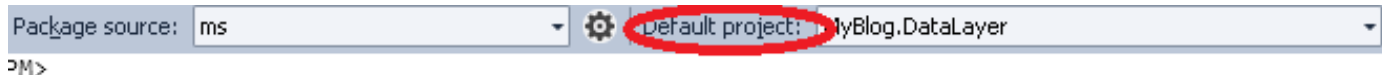
قدیمی هست نمونه موجود در آن. مگر اینکه از آخرین نگارش VS.NET به همراه آخرین به روز رسانی آن استفاده کنید. اطلاعات بیشتر در مطالب به روز رسانی به EF 6 : [اینجا](#) و [اینجا](#)
همچنین این پروژه به علت ذات سورس باز آن هر از چندگاهی مستقل از VS.NET به روز می‌شود. بنابراین همیشه آخرین نگارش آن را بهتر است [از نیوگت دریافت کنید](#) .

نویسنده: مجتبی فخاری
تاریخ: ۱۳۹۲/۱۱/۲۷ ۱۸:۱

با سلام
اگه یه پروژه باشه که دارای چند Class Library با نام های DataLayer و DomainClasses و ServiceLayer و Models باشه
چطوری با Package Manager Console می‌تونم EF را برای هر پروژه نصب کنم؟ و اینکه چطوری می‌تونم دستور Install-Package EntityFramework.Migrations -Version 0.9.0.0 را فقط در پروژه DataLayer نصب کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۲۷ ۱۸:۲۶

پروژه پیش فرض را در کنسول پاورشل نیوگت باید تغییر بدید:



نویسنده: مجتبی فخاری
تاریخ: ۱۳۹۲/۱۱/۲۷ ۲۱:۵۰

با سلام

آیا باید برای هر پروژه از دستور Install-Package EntityFramework استفاده کنم؟
راهی نداره که نخوایم برای هر پروژه ای اونرا دانلود و نصب نماییم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۲۷ ۲۲:۱۰

نیوگت خیلی بهینه عمل می کند. بار اول که درخواست نصب بسته ای را می دهید، ابتدا یک کوثری ساده برای دریافت شماره آخرین نگارش موجود در مخزن سایت رسمی آن ارسال می کند. بعد این شماره نگارش را با کش محلی سیستم (فایل های قبلی دریافت شده آن) مقایسه می کند. اگر یکی بود، از کش محلی استفاده می شود (چیزی مجددا دریافت نخواهد شد)؛ در غیر این صورت بسته ی جدید را دریافت خواهد کرد.

نویسنده: مجتبی فخاری
تاریخ: ۱۳۹۲/۱۱/۲۷ ۲۲:۱۰

وقتی سعی می کنم که از دستور زیر استفاده کنم با خطای زیر روبه رو می شوم. PM> Install-Package EntityFramework

```
Install-Package : Could not connect to the feed specified at 'https://www.nuget.org/api/v2/'. Please
verify that the package source
(located in the Package Manager Settings) is valid and ensure your network connectivity.
At line:1 char:1
+ Install-Package EntityFramework
+ ~~~~~
+ CategoryInfo          : NotSpecified: (:) [Install-Package], InvalidOperationException
+ FullyQualifiedErrorId : NuGetCmdletUnhandledException,NuGet.PowerShell.Commands.InstallPackageCommand
```

راه حل چیست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۲۷ ۲۲:۱۵

اینجا بحث شده: « [خلاصه ای در مورد روش های دریافت فایل از سایت NuGet](#) »

نویسنده: پژمان
تاریخ: ۱۳۹۲/۱۲/۳۰ ۱۰:۴۰

همانطور که شما فرمودید که کلاس Context را در یک ClassLibrary جداگانه قرار بدیم و نیاز به ارجاعی به اسمبلی های EF خواهد داشت. این یعنی من باید برای این ClassLibrary هم باید EF را بوسیله Nuget نصب کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۳۰ ۱۰:۴۳

[کمی بالاتر](#) در نظرات با تصویر پاسخ دادم.

نویسنده: مرتضی احمدی
تاریخ: ۱۵:۱۹ ۱۳۹۲/۱۲/۲۸

سلام

اگر در سازنده DbContext با مقدار ثابتی مثل "base" connectionName مقداردهی کنیم به کانکشن موردنظر وصل می‌شود ولی وقتی که این مقدار را Runtime ست می‌کنم عمل نمی‌کند و خطا میدهد کانکشن پیدا نشد. در حالی که معادل نامی که Runtime ست شده است کانکشن استرینگی تعریف شده است.

نویسنده: وحید نصیری
تاریخ: ۱۷:۱۰ ۱۳۹۲/۱۲/۲۸

تنظیم رشته اتصالی در زمان اجرا:

```
var ctx = new MyContext();
ctx.Database.Connection.ConnectionString = "...";
```

نویسنده: حمید حسین وند
تاریخ: ۱۳:۲۱ ۱۳۹۳/۰۱/۲۴

سلام

میخواهم بدونم فرق دو تا دستور زیر چیه با هم؟

```
public IList<Post> Posts { set; get; }
```

و

```
public ICollection<Post> Posts { set; get; }
```

و اینکه تا جایی که می‌دونم نباید فیلد اضافی به اسم این Property ها در table ایجاد بشه اما توی کدهایی که من نوشتم (عین همین دو مورد) برای هر کدوم فیلد اضافی توی جدولم ایجاد میشه و مقدارش null هست. مگر نه اینکه از این دو مورد برای دریافت اطلاعات اضافی از جدول مثلا Post استفاده میشه و لزومی برای درج اطلاعات هنگام ثبت وبلاگ جدید نیست؟

نویسنده: وحید نصیری
تاریخ: ۱۳:۳۱ ۱۳۹۳/۰۱/۲۴

در قسمت‌های بعدی به این مباحث پرداخته شده:

- فرق List و کالکشن و موارد مشابه [در اینجا](#)

- بررسی رابطه one-to-many [در قسمت هفتم](#)

نویسنده: رشوند
تاریخ: ۱۹:۴۷ ۱۳۹۳/۰۱/۳۱

سلام و با عرض تبریک روز زن به همه زنان ایران زمین.
با [پیشنهادتان](#) برای ارتباط با دیتابیس، این سری از آموزش‌ها رو شروع کردم.

سوال:

در قسمت تشکیل خودکار بانک اطلاعاتی و افزودن اطلاعات به جداول

1- مقدار Data Source و Initial Catalog رو از کجا باید پیدا کرد؟ یا بهتر بگویم کانکشن استرینگی رو چطوری می‌شه از SQL

بدست آورد؟

این کانکشن بعد از نصب SQL Server 2008 Enterprise :

Current connection properties:

Authentication Method: Windows Authentication
User Name: rashvand-PC\other

Connection

Database	master
SPID	54
Network Protocol	<default>
Network Packet Size	4096
Connection Timeout	15
Execution Timeout	0
Encrypted	No

Product

Product Name	Microsoft SQL Server Enterprise Evaluation Edition
Product Version	10.0.1600 RTM
Server Name	RASHVAND-PC
Instance Name	
Language	English (United States)
Collation	SQL_Latin1_General_CP1_CI_AS

Server Environment

Computer Name	RASHVAND-PC
Platform	NT INTEL X86
Operating System	6.1 (7601)
Processors	6
Operating System Memory	3327

User Name
The name of the user or login used to connect.

Close Help

2- ما در ابتدای آموزش ی اد گرفتیم که برای شروع کار یک کنترلر و بعد اکشن و بعد ویو ایجاد کنیم و سپس پروژه رو اجرا کنیم (در حال کلی). حالا می‌خواستم بپرسم که برای اجرای (در اولین بار اجرای کدهای زیر) کلاس Program رو چگونه (کجا و چگونه بنویسیم) اجرا بگیریم تا دیتابیس ایجاد شود..؟

سپاس.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۳۱ ۲۰:۴۵

- رشته اتصالی به SQL Server حالت‌های مختلفی می‌تواند داشته باشد. [اطلاعات بیشتر](#)
Data Source آن معمولاً نام کامپیوتر جاری است یا IP Server. چون در تصویر شما instance name خالی است، از همان وهله‌ی پیش فرض استفاده می‌شود. اگر مقدار داشت می‌شد computer_name/instance_name
Initial Catalog نام بانک اطلاعاتی مدنظر است که قرار است به آن متصل شوید (یا در اینجا به صورت خودکار ساخته شود).
Integrated Security = true به معنای استفاده از اعتبارسنجی ویندوزی است برای اتصال به SQL Server. یعنی کاربر جاری لاگین کرده به سیستم باید دسترسی لازم را برای کار با SQL Server داشته باشد.
- برای فراگیری یک فناوری جدید از برنامه‌های کنسول استفاده کنید و نه ASP.NET. این مباحث عمومی است بین فناوری‌های مختلف استفاده کننده از آن. در یک برنامه‌ی کنسول آغاز کار از متد Main است؛ در یک برنامه‌ی وب از متد Application_Start فایل global.asax.cs خواهد بود.

نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۰۹/۲۵ ۱۹:۴۱

در بعضی موارد هنگام کار با EF چنین خطایی رخ میدهد

Validation failed for one or more entities. See 'EntityValidationErrors' property for more details

البته اگر من از طریق زیر عمل کنم هیچ اتفاق نمیفته و درج انجام میشه

```
db.Entry(message).State = EntityState.Added;
db.SaveChanges();
```

حالا تفاوت این دو روش db.messages.add با db.entry چه تفاوتی داره؟
یکی سری کدها هم داخل نت برای catch کردن DbEntityValidationException وجود داره ولی هیچ وقت وارد catch نمیشه ، مثل اینکه نوع استثنا رخ داده متفاوته

نمونه کدهای موجود

```
try
{
    // Your code...
    // Could also be before try if you know the exception occurs in SaveChanges
    context.SaveChanges();
}
catch (DbEntityValidationException e)
{
    foreach (var eve in e.EntityValidationErrors)
    {
        Console.WriteLine("Entity of type \"{0}\" in state \"{1}\" has the following validation errors:",
            eve.Entry.Entity.GetType().Name, eve.Entry.State);
        foreach (var ve in eve.ValidationErrors)
        {
            Console.WriteLine("- Property: \"{0}\", Error: \"{1}\"",
                ve.PropertyName, ve.ErrorMessage);
        }
    }
    throw;
}
```

نویسنده: وحید نصیری
تاریخ: ۲۰:۵۲ ۱۳۹۳/۰۹/۲۵

البته این مطلب اول هست. در مطالب بعدی در مورد «اعتبارسنجی» بیشتر بحث شده:

[Entity Framework و InnerException](#)

[استثنائاتی که باید حین استفاده از EF Code first بررسی شوند](#)

[اعتبارسنجی در EF Code first](#)

همچنین نیاز است با «[رفتار متصل و غیر متصل در EF](#)» آشنا شوید. این مورد در مطلب «[کار با کلیدهای اصلی و خارجی در EF Code first](#)» هم بیشتر بحث شده است.

تمام این‌ها در [مسیر آموزشی EF Code first](#) به ترتیب لیست شده‌اند؛ قسمت‌های «اعتبارسنجی و بررسی استثناءها» و «ردیابی تغییرات».

نویسنده: امیر نوروزیان
تاریخ: ۸:۲۴ ۱۳۹۳/۱۰/۱۸

سلام؛ من می‌خام بعضی از مباحث سایت رو خروجی pdf بگیرم چیکار باید کنم مثلاً همین مبحث ef cf اگر جز امکانات مدیر سایت همین cf بی زحمت لینکش بهم بدین ممنون

نویسنده: وحید نصیری
تاریخ: ۹:۲۳ ۱۳۹۳/۱۰/۱۸

- «[دریافت خروجی کامل NET Tips](#)»

- برای مثال خروجی کامل بحث Entity Framework [در پوشه‌ی Tags](#) واقع شده: ([^](#))

- بانک اطلاعاتی سایت هم برای دریافت موجود است؛ به همراه Viewer آن: ([^](#))

- در پوشه‌ی [LearningPaths](#) ، خروجی‌های اختصاصی‌تری تهیه شده‌اند. برای مثال این خروجی اختصاصی و انتخابی EF Code First است: ([^](#))

نویسنده: پاییز
تاریخ: ۰:۴۴ ۱۳۹۳/۱۲/۱۴

جدول MigrationHistory_ را در اس کیو ال در فلدر System Tables قرار نداده است و آن را کنار دو جدول Posts و Blogs قرار داده است ؛ در صورتیکه در عکس شما این جدول به عنوان جدول سیستمی معرفی شده است. علت چیست؟

نویسنده: وحید نصیری
تاریخ: ۰:۵۱ ۱۳۹۳/۱۲/۱۴

چون کاربران زیادی با این مساله مشکل پیدا کرده بودند، در نگارش‌های اخیر ترجیح داده شده که این جدول سیستمی نباشد.

نویسنده: علی یگانه مقدم
تاریخ: ۱۵:۴۲ ۱۳۹۴/۰۵/۱۴

آیا در EF این امکان وجود دارد که بتوان filegroups ایجاد کرد و هر کدام از جداول را در filegroups مخصوص قرار داد؟ یا اینکه تنظیمات ابتدایی ساخت دیتابیس را برای آن تعیین کرد؟
مثلاً در sql server حجم ابتدایی دیتابیس و Filegrowth را تعیین کنیم؟
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۶:۱۹ ۱۳۹۴/۰۵/۱۴

بله. در متد **Up** مباحث [migrations](#) می‌توان [مستقیماً SQL](#) هم نوشت و آنرا سفارشی سازی کرد:

```
public override void Up()
{
    Sql(...);
    CreateTable(...);
    Sql(...);
}
```