عنوان: EF Code First #15
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۲/۳۰ ۱۲:۲۰:۰۰
آدرس: www.dotnettips.info
گروه‌ها: Entity framework

EF Code first و بانک‌های اطلاعاتی متفاوت

در آخرین قسمت از سری EF Code first بد نیست نحوه استفاده از بانک‌های اطلاعاتی دیگری را بجز SQL Server نیز بررسی کنیم. در اینجا کلاس‌های مدل و کدهای مورد استفاده نیز همانند قسمت 14 است و تنها به ذکر تفاوت‌ها و نکات مرتبط اکتفاء خواهد شد.

حالت کلی پشتیبانی از بانک‌های اطلاعاتی مختلف توسط EF Code first

EF Code first با کلیه پروایدرهای تهیه شده برای ADO.NET 3.5 که پشتیبانی از EF را لحاظ کرده باشند، به خوبی کار می‌کند.
پروایدرهای مخصوص ADO.NET 4.0، تنها سه گزینه DeleteDatabase/CreateDatabase/DatabaseExists را نسبت به نگارش قبلی بیشتر دارند و EF Code first ویژگی‌های بیشتری را طلب نمی‌کند.
بنابراین اگر حین استفاده از پروایدر ADO.NET مخصوص بانک اطلاعاتی خاصی با پیغام «CreateDatabase is not supported by the provider» مواجه شدید، به این معنا است که این پروایدر برای دات نت 4 به روز نشده است. اما به این معنا نیست که با EF Code first کار نمی‌کند. فقط باید یک دیتابیس خالی از پیش تهیه شده را به برنامه معرفی کنید تا مباحث Database Migrations به خوبی کار کنند؛ یا اینکه کلا می‌توانید Database Migrations را خاموش کرده (متد Database.SetInitializer را با پارامتر نال فراخوانی کنید) و فیلدها و جداول را دستی ایجاد کنید.

استفاده از EF Code first با SQLite

برای استفاده از SQLite در دات نت ابتدا نیاز به پروایدر ADO.NET آن است: « مکان دریافت درایورهای جدید SQLite مخصوص دات نت »
ضمن اینکه به نکته « استفاده از اسمبلی‌های دات نت 2 در یک پروژه دات نت 4 » نیز باید دقت داشت.
و یکی از بهترین management studio هایی که برای آن تهیه شده: « SQLite Manager »
پس از دریافت پروایدر آن، ارجاعی را به اسمبلی System.Data.SQLite.dll به برنامه اضافه کنید.
سپس فایل کانفیگ برنامه را به نحو زیر تغییر دهید:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <section name="entityFramework"
type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=4.3.1.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" />
  </configSections>
  <startup useLegacyV2RuntimeActivationPolicy="true">
    <supportedRuntime version="v4.0"/>
  </startup>

  <connectionStrings>
    <clear/>
    <add name="Sample09Context"
         connectionString="Data Source=CodeFirst.db"
         providerName="System.Data.SQLite"/>
  </connectionStrings>
</configuration>
```

همانطور که ملاحظه می‌کنید، تفاوت آن با قبل، تغییر connectionString و providerName است.

اکنون اگر همان برنامه قسمت قبل را اجرا کنیم به خطای زیر برخواهیم خورد:
«The given key was not present in the dictionary»
در این مورد هم توضیح داده شد. سه گزینه DeleteDatabase/CreateDatabase/DatabaseExists در پروایدر جاری SQLite برای دات نت وجود ندارد. به همین جهت نیاز است فایل «CodeFirst.db» ذکر شده در کانکشن استرینگ را ابتدا دستی درست کرد.
برای مثال از افزونه SQLite Manager استفاده کنید. ابتدا یک بانک اطلاعاتی خالی را درست کرده و سپس دستورات زیر را بر روی بانک اطلاعاتی اجرا کنید تا دو جدول خالی را ایجاد کند (در برگه Execute sql افزونه SQLite Manager):

```
CREATE TABLE [Payees](
 [Id] [integer] PRIMARY KEY AUTOINCREMENT NOT NULL,
 [Name] [text] NULL,
 [CreatedOn] [datetime] NOT NULL,
 [CreatedBy] [text] NULL,
 [ModifiedOn] [datetime] NOT NULL,
 [ModifiedBy] [text] NULL
);

CREATE TABLE [Bills](
 [Id] [integer] PRIMARY KEY AUTOINCREMENT NOT NULL,
 [Amount] [float](18, 2) NOT NULL,
 [Description] [text] NULL,
 [CreatedOn] [datetime] NOT NULL,
 [CreatedBy] [text] NULL,
 [ModifiedOn] [datetime] NOT NULL,
 [ModifiedBy] [text] NULL,
 [Payee_Id] [integer] NULL
);
```

سپس سطر زیر را نیز به ابتدای برنامه اضافه کنید:

```
Database.SetInitializer<Sample09Context>(null);
```

به این ترتیب database migrations خاموش می‌شود و اکنون برنامه بدون مشکل کار خواهد کرد.
فقط باید به یک سری نکات مانند نوع داده‌ها در بانک‌های اطلاعاتی مختلف دقت داشت. برای مثال integer در اینجا از نوع Int64 است؛ بنابراین در برنامه نیز باید به همین ترتیب تعریف شود تا نگاشت‌ها به درستی انجام شوند.

در کل تنها مشکل پروایدر فعلی SQLite عدم پشتیبانی از مباحث database migrations است. این مورد را خاموش کرده و تغییرات ساختار بانک اطلاعاتی را به صورت دستی به بانک اطلاعاتی اعمال کنید. بدون مشکل کار خواهد کرد.

البته اگر به دنبال پروایدری تجاری با پشتیبانی از آخرین نگارش EF Code first هستید، گزینه زیر نیز مهیا است:
/http://devart.com/dotconnect/sqlite
برای مثال اگر علاقمند به استفاده از حالت تشکیل بانک اطلاعاتی SQLite در حافظه هستید (با رشته اتصالی ویژه Data Source=:memory:;Version=3;New=True;)، فعلا تنها گزینه مهیا استفاده از پروایدر تجاری فوق است؛ زیرا مبحث Database Migrations را به خوبی پشتیبانی می‌کند.

استفاده از EF Code first با SQL Server CE

قبلا در مورد « استفاده از SQL-CE به کمک NHibernate » مطلبی را در این سایت مطالعه کرده‌اید. سه مورد اول آن با EF Code first یکی است و تفاوتی نمی‌کند (یک سری بحث عمومی مشترک است). البته با یک تفاوت؛ در اینجا EF Code first قادر است یک بانک اطلاعاتی خالی SQL Server CE را به صورت خودکار ایجاد کند و نیازی نیست تا آن‌را دستی ایجاد کرد. مباحث database migrations و به روز رسانی خودکار ساختار بانک اطلاعاتی نیز در اینجا پشتیبانی می‌شود.

برای استفاده از آن ابتدا ارجاعی را به اسمبلی System.Data.SqlServerCe.dll قرار گرفته در مسیر Program Files\Microsoft SQL Server Compact Edition\v4.0\Desktop اضافه کنید.
سپس رشته اتصالی به بانک اطلاعاتی و providerName را به نحو زیر تغییر دهید:

```
<connectionStrings>
    <clear/>
    <add name="Sample09Context"
         connectionString="Data Source=mydb.sdf;Password=1234;Encrypt Database=True"
         providerName="System.Data.SqlServerCE.4.0"/>
</connectionStrings>
```

بدون نیاز به هیچگونه تغییری در کدهای برنامه، همین مقدار تغییر در تنظیمات ابتدایی برنامه برای کار با SQL Server CE کافی است.
ضمنا مشکلی هم با فیلد Identity در آخرین نگارش EF Code first وجود ندارد؛ برخلاف حالت database first آن که پیشتر این اجازه را نمی‌داد و خطای «Server-generated keys and server-generated values are not supported by SQL Server Compact» را ظاهر می‌کرد.

استفاده از EF Code first با MySQL

برای استفاده از EF Code first با MySQL (نگارش 5 به بعد البته) ابتدا نیاز است پروایدر مخصوص ADO.NET آن‌را دریافت کرد: (^)
که از EF نیز پشتیبانی می‌کند. پس از نصب آن، ارجاعی را به اسمبلی MySql.Data.dll قرار گرفته در مسیر Program Files\MySQL\MySQL Connector Net 6.5.4\Assemblies\v4.0 به پروژه اضافه نمائید.
سپس رشته اتصالی و providerName را به نحو زیر تغییر دهید:

```
<connectionStrings>
    <clear/>
    <add name="Sample09Context"
         connectionString="Datasource=localhost; Database=testdb2; Uid=root; Pwd=123;"
         providerName="MySql.Data.MySqlClient"/>
</connectionStrings>

<system.data>
    <DbProviderFactories>
      <remove invariant="MySql.Data.MySqlClient"/>
      <add name="MySQL Data Provider"
           invariant="MySql.Data.MySqlClient"
           description=".Net Framework Data Provider for MySQL"
           type="MySql.Data.MySqlClient.MySqlClientFactory, MySql.Data, Version=6.5.4.0,
Culture=neutral, PublicKeyToken=c5687fc88969c44d" />
    </DbProviderFactories>
</system.data>
```

همانطور که مشاهده می‌کنید در اینجا شماره نگارش دقیق پروایدر مورد استفاده نیز ذکر شده است. برای مثال اگر چندین پروایدر روی سیستم نصب است، با مقدار دهی DbProviderFactories می‌توان از نگارش مخصوصی استفاده کرد.

با این تغییرات پس از اجرای برنامه قسمت قبل، به خطای زیر برخواهیم خورد:
The given key was not present in the dictionary

توضیحات این مورد با قسمت SQLite یکی است؛ به عبارتی نیاز است بانک اطلاعاتی testdb را دستی درست کرد. همچنین

جداول و فیلدها را نیز باید دستی ایجاد کرد و database migrations را نیز باید خاموش کرد (پارامتر Database.SetInitializer را به نال مقدار دهی کنید).
برای این منظور یک دیتابیس خالی را ایجاد کرده و سپس دو جدول زیر را به آن اضافه کنید:

```
CREATE TABLE IF NOT EXISTS `bills` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `Amount` float DEFAULT NULL,
  `Description` varchar(400) CHARACTER SET utf8 COLLATE utf8_persian_ci NOT NULL,
  `CreatedOn` datetime NOT NULL,
  `CreatedBy` varchar(400) CHARACTER SET utf8 COLLATE utf8_persian_ci NOT NULL,
  `ModifiedOn` datetime NOT NULL,
  `ModifiedBy` varchar(400) CHARACTER SET utf8 COLLATE utf8_persian_ci NOT NULL,
  `Payee_Id` int(11) NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_persian_ci AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `payees` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `Name` varchar(400) CHARACTER SET utf8 COLLATE utf8_persian_ci NOT NULL,
  `CreatedOn` datetime NOT NULL,
  `CreatedBy` varchar(400) CHARACTER SET utf8 COLLATE utf8_persian_ci NOT NULL,
  `ModifiedOn` datetime NOT NULL,
  `ModifiedBy` varchar(400) CHARACTER SET utf8 COLLATE utf8_persian_ci NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_persian_ci AUTO_INCREMENT=1 ;
```

پس از این تغییرات، برنامه بدون مشکل اجرا خواهد شد (ایجاد بانک اطلاعاتی خالی به همراه ایجاد ساختار جداول و خاموش کردن database migrations که توسط این پروایدر پشتیبانی نمی‌شود).

به علاوه پروایدر تجاری دیگری هم در سایت devart.com برای MySQL و EF Code first مهیا است که مباحث database migrations را به خوبی مدیریت می‌کند.

مشکل!
اگر به همین نحو برنامه را اجرا کنیم، فیلدهای یونیکد فارسی ثبت شده در MySQL با «??????? ?? ????» مقدار دهی خواهند شد و تنظیم CHARACTER SET utf8 COLLATE utf8_persian_ci نیز کافی نبوده است (این مورد با SQLite یا نگارش‌های مختلف SQL Server بدون مشکل کار می‌کند و نیاز به تنظیم اضافه‌تری ندارد):

```
ALTER TABLE `bills` DEFAULT CHARACTER SET utf8 COLLATE utf8_persian_ci
```

برای رفع این مشکل توصیه شده است که CharSet=UTF8 را به رشته اتصالی به بانک اطلاعاتی اضافه کنیم. اما در این حالت خطای زیر ظاهر می‌شود:

The provider did not return a ProviderManifestToken string

این مورد فقط به اشتباه بودن تعاریف رشته اتصالی بر می‌گردد؛ یا عدم پشتیبانی از تنظیم اضافه‌ای که در رشته اتصالی ذکر شده است.
مقدار صحیح آن دقیقا مساوی CHARSET=utf8 است (با همین نگارش و رعایت کوچکی و بزرگی حروف؛ مهم!):

```
<connectionStrings>
    <clear/>
    <add name="Sample09Context"
         connectionString="Datasource=localhost; Database=testdb; Uid=root; Pwd=123;CHARSET=utf8"
         providerName="MySql.Data.MySqlClient"/>
</connectionStrings>
```

به این ترتیب، مشکل ثبت عبارات یونیکد فارسی برطرف می‌شود (البته جدول هم بهتر است به DEFAULT CHARACTER SET utf8 COLLATE utf8_persian_ci تغییر پیدا کند؛ مطابق دستور Alter ایی که در بالا ذکر شد).

نظرات خوانندگان

نویسنده: MehdiPayervand
تاریخ: ۱۳۹۱/۰۲/۳۰ ۱۲:۴۱:۵۱

اول اینکه دستتون درد نکنه، واقعا سری هایی که شما برای اشتراک دانشتون توی بلاگ میذاریم بروز و کارآمد هستند، امیدوارم بتونیم به بهترین نحو از این دانسته ها استفاده کنیم.
و دانشمون اونقدری بشه که به اشتراک گذاشت (;

نویسنده: NTC
تاریخ: ۱۳۹۱/۰۲/۳۰ ۲۰:۳۶:۱۴

مطالب کامل و جامعی رو نوشتید.
از زحمات بی دریغتون کمال تشکر را دارم.
اما همچنان منتظر مطالب دیگر هم هستیم.

نویسنده: فرشید ابراهیمی
تاریخ: ۱۳۹۱/۰۴/۳۰ ۱۸:۳۷

اگر امکان دارد نحوه اتصال به اراکل را نیز توضیح دهید

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۳۰ ۱۸:۴۷

تعداد بانک‌های اطلاعاتی مهیا خیلی زیاد است. اگر این قسمت را مرور کرده باشید، حدود کار دستتان آمده است و این مهم است. چه خطاهایی را ممکن است دریافت کنید. راه حل چیست. پروایدر مخصوص نیاز دارید که احتمالا در سایت مربوطه قابل دسترسی است و از این دست موارد.
برای نمونه پروایدر رسمی Oracle ODP.Net با EF Code first کار می‌کند و یا یک نمونه دیگر در اینجا (^)

نویسنده: رضا
تاریخ: ۱۳۹۱/۰۶/۰۴ ۱۹:۹

میخواستم بدونم EntityFramework.SqlServerCompact که در Nuget هستش چه تفاوتی با Entity Framework معمولی داره؟
من دیتابیسم Sql Ce هستش. یعنی استفاده از این Package بهتر هستش؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۰۴ ۱۹:۲۰

سورس کد EF رو که دریافت کنید، یک پوشه به نام EntityFramework.SqlServerCompact داخل آن هست. به عبارتی آخرین نگارش EF به همراه پروایدر توکار SQL CE هم هست (و بوده). ضمنا این پروایدر به تنهایی کار نمی‌کند و نیاز خواهید داشت که پروایدر ADO.NET مربوط به SQL CE را هم به پروژه اضافه کنید .

نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۲/۰۸/۱۸ ۱۳:۵۹

لطفا پروایدر MySQL مخصوص ADO.NET را که در وب سایت رسمی MySQL قابل دریافت نیست،در صورتی که براتون مقدور بود (حجم و ...) در اینجا بارگذاری کنید
با تشکر

نویسنده: وحید نصیری

تاریخ: ۱۳۹۲/۰۸/۱۸ ۱۴:۱۶

- لینک اصلی: http://cdn.mysql.com/Downloads/Connector-Net/mysql-connector-net-6.7.4.msi
- لینک کمکی در اینجا
- جزئیات نحوه استفاده از آن توسط یکی از اعضای تیم EF در اینجا: Entity Framework on MySQL

نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۲/۰۸/۱۸ ۱۴:۲۷

بسیار عالی / با سپاس

نویسنده: سوین
تاریخ: ۱۳۹۲/۱۰/۰۴ ۲۳:۴۷

با سلام؛ در دیتابیس Sql Server برای اینکه اطلاعات در صورت Valid بودن از Log فایل حذف بشه می‌یابیم و دیتابیس رو به صورت Simple قرار می‌دیم تا اندازه فایل Log افزایش پیدا نکنه و دستورش هم به این صورته

```
ALTER DataBase DBName SET RECOVERY SIMPLE
```

حالا این رو در ORMها و بطبع در EF Code First چطور میشه پیاده سازی کرد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۰۵ ۰:۱۱

- امکان « استفاده مستقیم از عبارات SQL در EF Code first » وجود دارد؛ برای تمام حالاتی که EF آن‌ها را به صورت توکار پشتیبانی نمی‌کند.
- محل قرار دادن تنظیمات مقدماتی از این دست، در متد Seed مهاجرت هست. یک مثال و مثالی دیگر در مورد کار با Seed.

نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۲/۱۱/۲۹ ۱۷:۹

نمونه‌ای از پیاده سازی اتصال به اوراکل 11g در Entity Framework 6 بوسیله پروایدر تجاری شرکت devart :

ابتدا نسخه آزمایشی dotconnect for oracle 8.2 professional را از این آدرس دریافت و آن را نصب می‌کنیم.

نصب آخرین نسخه Entity Framework از طریق پاور شل نیوگت.
افزودن Devart.Data.Oracle و Devart.Data.Oracle.Entity به Solution.
حذف تگ defaultConnectionFactory در entityFramework.
افزودن تگ زیر در قسمت providers همانند کد زیر:

```
<provider invariantName="Devart.Data.Oracle"
type="Devart.Data.Oracle.Entity.OracleEntityProviderServices, Devart.Data.Oracle.Entity,
Version=8.2.100.6, Culture=neutral, PublicKeyToken=09af7300eec23701" />
```

تکمیلی: اصول کلی دسترسی به اوراکل به شرح بالاست، ولی نکته مهم مقداردهی به خصیصه Version=X.X.X.X با توجه به نسخه اسمبلی Devart.Data.Oracle.Entity می‌باشد.

dotConnect for Oracle

dotConnect for Oracle