عنوان: **استفاده از پروایدر SQLite در Entity Framework 7**
نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۳/۰۸/۲۶ ۱۹:۳۰
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, SQLite, Entity Framework 7

Entity Framework در نگارش 7 خود از منابع داده‌ایی جدیدی پشتیبانی میکند(+) . یعنی از Windows Store، Windows Phone و همچنین ASP.NET 5 (اپلیکیشن‌هایی که از .NET Core استفاده می‌کنند) پشتیبانی خواهد کرد. در این نسخه از دیتابیس‌های non-relational نیز پشتیبانی می‌شود. پروایدر SQLite به صورت رسمی توسط تیم EF ارائه شده است که در ادامه نحوه‌ی استفاده از آن را در یک برنامه کنسول ساده بررسی خواهیم کرد.
کلاس‌های برنامه:

```
using Microsoft.Data.Entity;
using Microsoft.Data.Entity.Metadata;
using System.Collections.Generic;
using System.Linq;

namespace UsingEF7WithSQLite
{
    public class Blog
    {
        public int BlogId { get; set; }
        public string Url { get; set; }

        public List<Post> Posts { get; set; }
    }

    public class Post
    {
        public int PostId { get; set; }
        public string Title { get; set; }
        public string Content { get; set; }

        public int BlogId { get; set; }
        public Blog Blog { get; set; }
    }
}
```

خب تا اینجا مدل‌های برنامه را تعریف کردیم، قدم بعدی افزودن پکیج مربوط به پروایدر SQLite به پروژه است، با دستور زیر این پکیج را نصب می‌کنیم:

```
PM> Install-Package EntityFramework.SQLite –Pre
```

اکنون کلاس کانتکست برنامه را به صورت زیر تعریف می‌کنیم:

```
namespace UsingEF7SQLiteProvider
{
    public class BloggingContext : DbContext
    {
        public DbSet<Blog> Blogs { get; set; }
        public DbSet<Post> Posts { get; set; }

        protected override void OnConfiguring(DbContextOptions builder)
        {
            builder.UseSQLite(@"Data Source=.\BloggingDatabae.db");
        }

        protected override void OnModelCreating(ModelBuilder builder)
        {
            builder.Entity<Blog>()
                .OneToMany(b => b.Posts, p => p.Blog)
                .ForeignKey(p => p.BlogId);

            // The EF7 SQLite provider currently doesn't support generated values
            // so setting the keys to be generated from developer code
            builder.Entity<Blog>()
                .Property(b => b.BlogId)
                .GenerateValueOnAdd(false);
```

```
            builder.Entity<Post>()
                .Property(b => b.BlogId)
                .GenerateValueOnAdd(false);
        }
    }
}
```

کار را با بازنویسی متد OnConfiguration شروع می‌کنیم، در این قسمت باید به EF بگوئیم که می‌خواهیم از SQLite استفاده کنیم برای اینکار از یک Extension Method با نام UseSQLite و پاس دادن کانکشتن استرینگ به آن استفاده می‌کنیم. **نکته:** پروایدر فعلی SQLite در حال حاضر از Generated values پشتیبانی نمی‌کند، برای این منظور باید درون متد OnModelCreating این قابلیت را غیرفعال کنیم.
اکنون می‌توانیم از طریق پاورشل نیوگت دیتابیس را ایجاد کنیم، برای اینکار باید پکیج زیر را به پروژه اضافه کنید:

```
Install-Package EntityFramework.Commands -Pre
```

سپس دستورات زیر را اجرا می‌کنیم:

```
Add-Migration MyFirstMigration
Apply-Migration
```

توسط دستور Apply-Migrate دیتابیس برای شما ایجاد خواهد شد. البته این دستور زمانی استفاده می‌شود که برنامه شما یک اپلیکیشن دسکتاپ باشد. اگر اپلیکیشن شما یک Windows Phone Application است باید در زمان اجرای برنامه این کد را بنویسید:

```
using (var db = new BloggingContext())
{
   db.Database.AsMigrationsEnabled().ApplyMigrations();
}
```

در نهایت می‌توانید با دستور زیر از کانتکست برنامه استفاده کرده و خروجی را مشاهده کنید:

```
using (var db = new Models.BloggingContext())
{
     db.Blogs.Add(new Models.Blog { Url = "http://dotnettips.info" });
     db.SaveChanges();

     foreach (var item in db.Blogs)
     {
         Console.WriteLine(item.Url);
     }
}
Console.ReadLine();
```