عنوان: ObservableCollection در Entity Framework
نویسنده: شهروز جعفری
تاریخ: ۱۳۹۱/۰۵/۰۱ ۱۸:۵۷
آدرس: www.dotnettips.info
برچسب‌ها: Entity framework, DbContext

در مبحث ا ستفاده از خاصیت Local در Entity Framework ملاحظه نمودید که خاصیت Local به راحتی می‌تواند از رفت و آمدهای بی جهت به دیتابیس جلوگیری کند.
حال قصد معرفی یک collection را به نام ObservableCollection دارم.
همانطور که از نامش پیداست برای مشاهده و تحت نظر قرار دادن داده‌های اضافه شده یا پاک شده کاربرد دارد. به کد زیر دقت کنید.

```
private static void ListenToLocalChanges()
    {
        using (var context = new BreakAwayContext())
        {
            context.Destinations.Local.CollectionChanged += (sender, args) =>
            {
                if (args.NewItems != null)
                {
                    foreach (Destination item in args.NewItems)
                    {
                        Console.WriteLine("Added: " + item.Name);
                    }
                }
                if (args.OldItems != null)
                {
                    foreach (Destination item in args.OldItems)
                    {
                        Console.WriteLine("Removed: " + item.Name);
                    }
                }
            };
            context.Destinations.Load();
        }
    }
```

در بالا به وسیله یک event handler جدید به collection محلی ما (Local) نظر می‌اندازد و در صورت اضافه شدن یا حذف موجودیتی، آن را به ما نشان می‌دهد. فقط توجه کنید که اگر نیاز دارید در صفحه‌ای این تغییرات را مشاهده کنید باید عمل Refresh کردن صفحه را چه به صورت دستی یا با نوشتن کد خودتان مدیریت کنید. البته با استفاده از WPF میتوان (استفاده از کنترل‌های مانند ListBox) این کار را به صورت خودکار انجام داد.

نظرات خوانندگان

نویسنده: محمد زارع
تاریخ: ۱۳۹۱/۰۵/۰۲ ۰:۱۴

سلام آقای جعفری. ممنون از مطلب مفیدتون. شاید سوالم خیلی ربطی به مطلب شما نداشته باشه ولی میخواستم ازتون بپرسم که ما وقتی توی پروژه WPF از Entity Framework Code First استفاده میکنیم باید اینترفیس INotifyPropertyChanged رو برای هر Entity پیاده سازی کنیم یا نه؟!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۲ ۸:۱۵

- بله. باید اینکار را انجام دهید.
- استفاده مستقیم از مدل‌های EF در WPF یا MVC یا هر جای دیگری کار توصیه شده‌ای نیست.
View شما باید Model مخصوص به خود را داشته باشد و این مدل الزاما با موجودیت‌های بانک اطلاعاتی شما یکی نیست.
برای نگاشت اطلاعات مدل یک View به مدل داده‌ای می‌شود از کتابخانه‌هایی مانند Auto-Mapper استفاده کرد.

نویسنده: ایلیا
تاریخ: ۱۳۹۱/۰۶/۲۷ ۱۴:۵۹

با سلام.
در پروژه WPF در لایه سرویس یکبار Local رابر میگردانم مانند زیر :

```
public override IList<City> GetAll()
        {
            var query = from item in _tEntities
                        select item;
            query.Load();
            return _tEntities.Local;
        }
```

همه چیز درست است ولی وقتی برای جستجو متد زیر را اجرا می‌کنم باز Local شامل همان داده‌های قبلی است:

```
public override IList<City> GetAll(Func<City, bool> predicate)
        {
            var query = from item in _tEntities
                        select item;
            query.Where<City>(predicate);
            query.Load();
            return _tEntities.Local;
        }
```

لطفاً راهنمایی کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۷ ۱۶:۱۴

در مثال رسمی Code first EF و WPF یک سری Refresh هم هست که باید وقت بگذارید و مطالعه کنید.

نویسنده: kianush
تاریخ: ۱۳۹۱/۰۸/۰۳ ۱۳:۲۹

من در winform یک BindingSource دارم و این رو به گرید می‌دم. گرید به یک BindingSource بایند شده، اگه مقادیری رو تغییر بدم یا اضافه کنم در گرید (در واقع به BindingSource پشت صحنه) ، ChangeTracker تشخیص می‌ده و می‌تونم کار رو هندل کنم

اما اگه سطری رو از BindingSource (یا همون گرید) حذف کنم ChangeTracker متوجه نمی‌شه و وضعیتشون Unchanged می‌مونه! (Local هم فایده نداره) و نمی‌تونم کل تغییرات رو ذخیره کنم با صدا زدن SaveChanges
در ChangeTracker ایتم هایی با Stateهای Add, Modified رو می‌تونم ببینم ولی اگه ایتمی رو Delete کنم نمی‌تونم ببینمش و کاری انجام بدم! مشکل از کجاست؟
اینا مربوط به Context هست برای بررسی وضعیت تغییرات:

```
public bool HasChanges()
        {
            return this.ChangeTracker.Entries().Any(e => e.State != EntityState.Unchanged);
        }

        public void RejectChanges()
        {
            foreach (var entry in this.ChangeTracker.Entries())
            {
                switch (entry.State)
                {
                    case EntityState.Modified:
                        entry.State = EntityState.Unchanged;
                        break;

                    case EntityState.Added:
                        entry.State = EntityState.Detached;
                        break;
                }
            }
        }
```

توضیحات بیشتر:
چرا مثل Add , Modify عمل Delete در Entityها و ChangeTracker بصورت خودکار شنیده نمی‌شه؟! من یک کتابخانه‌ی مستقل تهیه کردم که گرید ،خودش یک کنترل BindingSource داره و اصلا نباید به EntityFramework و سرویسهایی که با Entityها کار می‌کنن در ارتباط باشه ، BindingSource باید بتونه CurrentItem اش رو Remove کنه در حافظه مثل کاری که برای Add,Update انجام می‌ده و در انتها من (بعنوان کاربر نهایی) در لایه نمایش تصمیم بگیرم که تمام تغییرات انجام شده در گرید (Add, Modify, Delete) رو ذخیره یا لغو کنم.

****** شاید بشه با کمک رویداد BindingSource.ListChanged به نحوی حل کرد ولی اصلا جالب نمی‌شه چون BindingSource من یک POCO Entity هست و فقط و فقط چنتا property داره و اصلا از وضعیت خودش خبر نداره و قرار هم نیست بیشتر از این باشه و من نمی‌تونم وضعیتش رو تغییر بدم در این رویداد چون همچین چیزی نداره اصلا!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۰۳ ۱۴:۰

- در مورد Tracking یک مطلب جداگانه در سایت هست. Tracking API همان ObservableCollection نیست. Tracking API در سایر ORMها نامی به شکل سطح اول کش دارد (first level caching).
- با توجه به اینکه برای بررسی کارهای شخصی و کتابخانه‌های مستقل، نیاز به کد کامل هست، بهتر است به مقاله زیر مراجعه کرده و جزئیات کار خودتان را با آن مقایسه کنید:
« Implementing Undo/Redo feature for DbContext of Entity Framework »

نویسنده: kianush
تاریخ: ۱۳۹۱/۰۸/۰۳ ۱۴:۳

ممنون.. بررسی می‌کنم ببینم می‌تونم اصولی حل کنم این مسئله رو یا خیر :-؟
ولی یه نکته ، اینکه گفتین "نیاز به کد کامل هست.." اصلا تصور کنین کتابخانه‌ی مستقلی نیست. مثلا یک Form, BindingSource, DataGridView رو داشته باشین و روال بالا که توضیحشو دادم. انگار یک Bug هست! یجورایی که وضعیت "حذف" رو مثل "افزوده شدن" و "تغییر کرده" نمی‌تونه اعلام کنه به دیتاسورس پشت سرش bindingsource

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۰۳ ۱۴:۲۳

- Tracking API فقط داخل یک Context مفهوم پیدا می‌کند نه مجزای آن.
- همچنین این API دارای متد DetectChanges هم هست که می‌شود به صورت دستی جهت اطمینان بیشتر آن‌را در هر زمانی (مثلا داخل بررسی HasChanges) فراخوانی کرد.

نویسنده: kianush
تاریخ: ۱۳۹۱/۰۸/۰۳ ۱۶:۶

برای این مسئله‌ی من راه حل اصولی ای پیدا نکردم. یه راهی که الان پیاده کردم و جواب گرفتم ولی جالب نیست: در BaseEntity پراپرتی IsDeleted رو کار گذاشتم مثلا یه همچین چیزی فک کنین:

```
public abstract class BaseEntity
    {
        [ColumnInfo("کد",pWidth:70)]
        public int Id { get; set; }

        [ColumnInfo("",pIsVisible:false,pIsEditable:false)]
        [NotMapped]
        public bool IsDeleted { get; set; }
    }
```

و جایی که BindingSource CurrentItem پاک می‌شه ، BindingSource.Current.IsDeleted=true (بصورت dynamic) گذاشتم و در 2 3 ، Context خط دیگه اضافه کردم که این رو هندل کنم برای تمام موجودیت ها.. کار می‌کنه ولی بدیش اینه که یک پراپرتی بی ربط (شاید به نوعی) رو در BaseEntity و در واقع در تمام موجودیتهام تعریف کردم (که البته NotMapped هست) و "رفتار" رو با "خاصیت" قاطی کردم و الان هم عذاب وجدان دارم :دی پ.ن: کماکان دنبال راهی می‌گردم با خوندن مقالات