عنوان: **مقدمه ای بر AutoMapper**
نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۴/۱۳ ۱۷:۲۲
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net MVC, AutoMapper, ASP.Net, MVC

AutoMapper کتابخانه ای ساده و سبک برای نگاشت اطلاعات یک شی به شی دیگر به صورت خودکار هست و...
اگه این پست رو مطالعه کرده باشید یه مشکل امنیتی بنام «Mass Assignment» مطرح شد.برای رفع این مشکل یک روش استفاده از ViewModel بود.
فرض کنید Model ما

```
public class User
    {
        public int Id { get; set; }

        public string FirstName { get; set; }

        public string LastName { get; set; }

        public string UserName { get; set; }

        public string Password { get; set; }

        public bool IsAdmin { get; set; }

        public virtual ICollection<BlogPost> BlogPosts { get; set; }
    }
```

و ViewModel ما

```
public class UserViewModel
    {
        public string FirstName { get; set; }

        public string LastName { get; set; }

        public string Password { get; set; }
    }
```

باشه(توجه کنید در واقع من برای View ی مورد نظرم فقط به نام , نام خانوادگی و پسورد نیاز دارم)
برای استفاده UserViewModel بعنوان Model در View ی مورد نظر باید شی UserViewModel رو با اطلاعات شی User مقدار دهی کنیم مثلا با کدی مثل این در کنترلر.

```
public ActionResult Index(int id = 1)
        {
            var user = _userService.GetById(id);
            var userViewModel = new UserViewModel
                {
                    FirstName = user.FirstName,
                    LastName = user.LastName,
                    Password = user.Password
                };

            return View(userViewModel);
        }
```

رهایی از نوشتن اینجور کدهای تکراری و خسته کننده باعث پیدایش AutoMapper شد...
برای استفاده از AutoMapper از نوگت استفاده میکنیم.

```
PM> Install-Package AutoMapper
```

در شروع برنامه نگاشت‌ها رو تعریف میکنم.یک روش ابداعی تعریف نگاشت‌ها در یک کلاس استاتیک و فراخوانی اون تو متد Application_Start هست.

```
public static class AutoMapperWebConfiguration
    {

        public static void Configure()
        {
            ConfigureUserMapping();
        }

        private static void ConfigureUserMapping()
        {
            Mapper.CreateMap<User, UserViewModel>();
        }
    }
```

اولین پارامتر نوع مبدا و دومین پارامتر نوع مقصد هست.
برای انجام نگاشت هم از متد Map استفاده میکنیم.

```
public ActionResult Index(int id=1)
        {
            var user = _userService.GetById(id);
            var userViewModel=new UserViewModel();
            AutoMapper.Mapper.Map(user, userViewModel);

            return View(userViewModel);
        }
```

همنطور که میبنید با نوشتن چندین خط کد عملیات نگاشت اطلاعات یک شی به شی دیگه انجام شد.
ادامه دارد...

نظرات خوانندگان

نویسنده: رضا
تاریخ: ۱۳۹۱/۰۴/۱۳ ۱۷:۳۲

دمت گرم مدتیه معطل همین مپینگ‌ها به صورت دستی هستم

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۳ ۱۷:۳۳

سلام
ممنون،
راه حل ساده واقعا لذت بخش هست...

نویسنده: daniel
تاریخ: ۱۳۹۱/۰۴/۱۳ ۲۱:۱۳

تو Compact Framwork از valueinjecter باید استفاده کرد.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۴/۱۳ ۲۳:۲

خیلی ممنونم ... میشه این سمپل رو ضمیمه کنید؟!

نویسنده: رضا.ب
تاریخ: ۱۳۹۱/۰۴/۱۳ ۲۳:۴۶

تشکر بابت معرفی.
کاربرد این Auto Mapper در اینجور نگاشت‌ها معنی میده فقط؟ یعنی نگاشت یه شیِ جزء به شیِ کل؟ یا مثلا کاربردهایی مثل ORM هم میتواند داشته باشد؟

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۴/۱۴ ۹:۲۱

یکی از مهمترین کاربردهاش نگاشت از یک مدل بزرگ و پیچیده به یک مدل ساده هست که بهش Flattening میگن و کاربردهای دیگه مثلا Projection و... که همنطور که آخر پست گفتم تو پست‌های بعدی تکمیلش میکنم.
برای مطالعه بیشتر همه به این ویکی مراجعه کنید.

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۴/۱۴ ۱۲:۱۶

سمپل خاصی نیست کدها رو از یک پروژه کشیدم بیرون...
تنها قسمتی که تو این پست نذاشتم(و البته نیاز هم نبود) View ش بود.

```
@model DotnetDevBlog.Web.Models.UserViewModel

@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>
@Html.DisplayForModel()
```

نویسنده: سعید یزدانی
تاریخ: ۱۳۹۲/۰۴/۰۲ ۱۶:۱۹

با سلام
ممنون از مطلب مفیدتون
میخواستم بدونم اگه از چند model بخواهیم استفاده کنیم باید چکار کنیم؟

نویسنده: محمد صاحب
تاریخ: ۱۳۹۲/۰۴/۰۴ ۱۰:۵۸

از این کلاس کمکی استفاده کن

نویسنده: hossein101211
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۳:۵۸

با سلام میخواستم ببینم آیا امکان استفاده از automapper بدین صورت هم وجود داره؟

```
public void Edit_news_ajax(News news)
        {
            using (var Context = new ProCamContext())
            {
                var q=Context.News.Find(news.id);
                AutoMapper.Mapper.Map(news,q);
               //NewsRepository.EditNews(news.id, q);
                Context.SaveChanges();
            }//end news
        }
```

با تشکر

نویسنده: محمد صاحب
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۴:۳۰

اگه q از نوع news هست که نیاز ی به AutoMapper نداری !

```
AutoMapper.Mapper.Map(news,q);
```

این خط مشکلی نداره

نویسنده: hossein101211
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۵:۵

من میخوام محتویات news رو (که توسط model binding پر شده) رو در سطری از جدولم بریزم (با اون آیدی که با find پیدا کردم جایگزین بشه) اینکار با AutoMapper امکان پذیر هست؟
برای اینکار نیاز به create.map هم دارم؟
با تشکر

نویسنده: محمد صاحب
تاریخ: ۱۳۹۲/۰۴/۱۱ ۱۱:۲۷

اگه نوع هاشون متفاوته بله باید Mapper.CreateMap رو تعریف کنی.

عنوان: **وی‌ژگی های پیشرفته ی AutoMapper - قسمت اول**
نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۴/۲۲ ۱۳:۴۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net MVC, AutoMapper

در پست قبلی مقدمه‌ای داشتیم بر AutoMapper؛ مثالی که در اون پست عنوان شد ساده‌ترین و پرکاربردترین روش استفاده از AutoMapper هست بنام Flattening که در واقع از یک شیء کل به یک شیء کوچکتر می‌رسیم.
همانطور که در قسمت اول گفتم AutoMapper کارش رو بر اساس قراردادها انجام میده یا همون Convention Base. یکی از قردادهای AutoMapper، نگاشت براساس نام اعضای اون شی هست؛ مثلا در مثال قبلی FirstName در مبداء، به خاصیتی با همین نام نگاشت شد و ...

Projection

روش استفاده بعدی که به اون Projection (معنی فارسی خوب برا Projection چیه ؟) میگن برای مواقعی هست که اعضای یک شیء در مبداء، همتایی در مقصد ندارد (از نظر نام) و در واقع می‌خواهیم یک شیء رو به یک شیء دیگه تغییر شکل بدیم.
مدل زیر رو در نظر بگیرید:

```
public class Person
    {
        public int Id { get; set; }

        public string FirstName { get; set; }

        public string LastName { get; set; }

    }
```

که قرار است به مدل PersonDTO نگاشت بشه.

```
public class PersonDTO
    {
        public int Id { get; set; }

        public string Name { get; set; }

        public string Family { get; set; }

public string FullName { get; set; }

        public string Email { get; set; }
    }
```

همنطور که می‌بینید در مقصد خاصیت‌هایی داریم که در مبداء همتایی ندارند. برای نگاشت چنین اشیایی، از متد ForMember استفاده و نگاشت‌های شی‌های موردنظر رو Custom می‌کنیم.

```
Mapper.CreateMap<Person, PersonDTO>().ForMember(des => des.Name, op => op.MapFrom(src =>
src.FirstName)).
                  ForMember(des => des.FullName, op => op.MapFrom(src => src.FirstName + " " +
src.LastName)).ForMember(
                      des => des.Email, op => op.Ignore());
```

متد ForMember از یک Action Delegate برای کانفیگ کردن هر عضو استفاده میکنه. در مثال بالا ما از MapFrom برای اعمال نگاشت Custom استفاده کردیم.

```
AutoMapper.IMappingExpression<TSource,TDestination>.ForMember(System.Linq.Expressions.Expression<System
.Func<TDestination,object>>, System.Action<AutoMapper.IMemberConfigurationExpression<TSource>>)
```

نکته :برای تست کردن اینکه آیا نگاشت ما Exception داره یا نه میتوان از متد زیر استفاده کرد.

```
Mapper.AssertConfigurationIsValid();
```

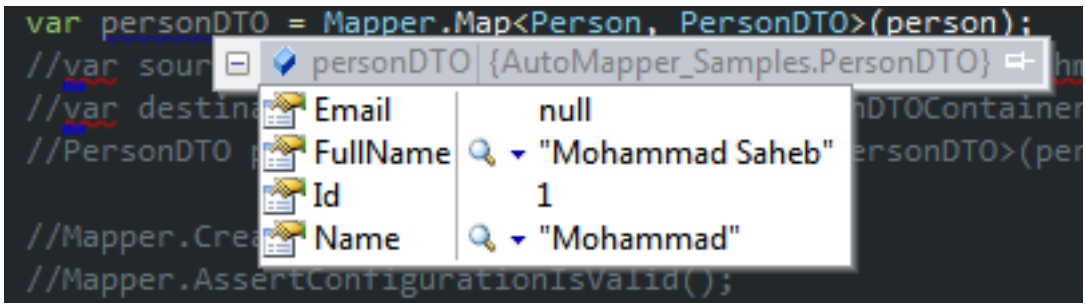
نکته : همیشه موقع نگاشت، اعضای شیء مقصد برای AutoMapper مهمن؛ مثلا در مثال بالا خاصیت LastName در مبداء به هیچ عضوی در مقصد نگاشت نشده (به صورت مستقیم) و این تولید Exception نمیکنه ولی برعکس اون باعث تولید Exception میشه؛ مثلا اگه خاصیت Email رو که در مبداء همتایی نداره رو کانفیگ نکنیم، باعث تولید Exception میشه.

نحوه استفاده

```
var person = new Person
                {
                    Id = 1,
                    FirstName = "Mohammad",
                    LastName = "Saheb",

                };
var personDTO = Mapper.Map<Person, PersonDTO>(person);
```

خروجی به شکل زیر میشه



Collection

بعد از نوشتن کانفیگ نگاشت‌ها، بدون نیاز به تنظیمات خاصی میتونیم مجموعه ای از شی‌های مبداء رو به مقصد نگاشت کنیم.
مجموعه‌های پشتیبانی شده به شرح زیرن.

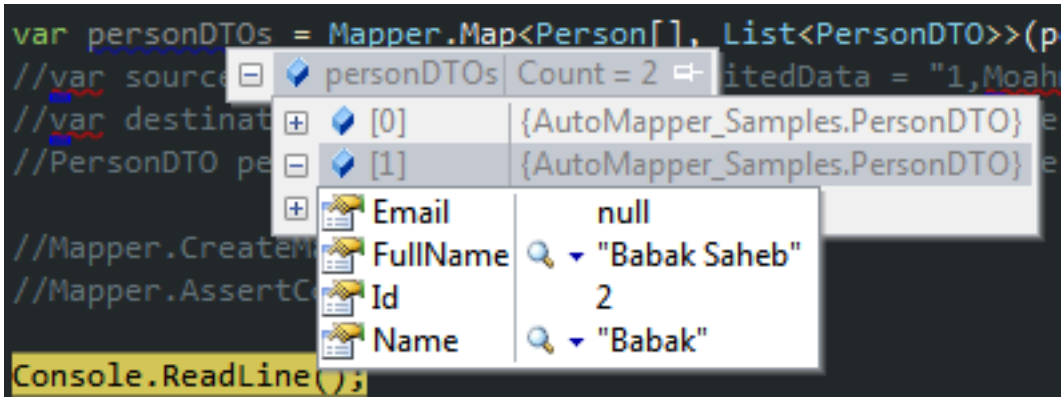
```
IEnumerable, IEnumerable<T>, ICollection, ICollection<T>, IList, IList<T>, List<T>.
```

و البته آرایه ها.
مثال

```
var persons = new[]
                {
                    new Person { Id = 1, FirstName = "Mohammad", LastName = "Saheb" },
                    new Person { Id = 2, FirstName = "Babak", LastName = "Saheb" }
                };

var personDTOs = Mapper.Map<Person[], List<PersonDTO>>(persons);
```

خروجی به شکل زیر میشه



Nested Mappings

برای نگاشت کلاس‌های تو در تو از این روش استفاده می‌کنیم و ...

فرض کنید در کلاس Person خاصیتی از نوع Address داریم و در کلاس PersonDTO خاصیتی از نوع AddressDTO.

```
public class Address
    {
        public string Ad1 { get; set; }
        public string Ad2 { get; set; }
    }


public class AddressDTO
    {
        public string Ad1 { get; set; }
        public string Ad2 { get; set; }
    }
```

برای نگاشت‌هایی از این دست باید تنظیمات نگاشت مربوط به نوع‌های تودرتو را صریحا معین کنیم.

```
Mapper.CreateMap<Address, AddressDTO>();
```

نکته: هرچند منطقی‌تر بنظر میرسه که تعریف نگاشت‌های داخلی ابتدا بیاد، ولی فرقی نمیکنه که تعریف این نگاشت قبل یا بعد از نگاشت اصلی باشه.

ادامه دارد...

نظرات خوانندگان

نویسنده: ابراهیم
تاریخ: ۱۳۹۱/۰۴/۲۲ ۲۱:۴۱

ممنون از مطالب خوبتان. Projection را به صورت پرتو در درس پایگاه دادهٔ استاد حق‌جو شنیده‌ام و به نظرم ترجمهٔ خوبی است.

نویسنده: فرهاد یزدان پناه
تاریخ: ۱۳۹۲/۰۳/۱۰ ۱:۳

وقت بخیر جناب صاحب
در حالاتی که از AutoMapper در یک عبارت LINQ استفاده بشه در صورت وجود مواردی همچون FullName در مقصد خطای "Object reference not set to an instance of an object" رو throw می‌کنه.
آیا شما هم با چنین مشکلی برخورد کردید؟

نویسنده: محمد صاحب
تاریخ: ۱۳۹۲/۰۳/۱۱ ۱۰:۴۶

نمیدونم شاید.برای برسی بیشتر کدهاش رو بذار

نویسنده: لیلا
تاریخ: ۱۳۹۲/۰۹/۰۴ ۲۳:۱۶

با تشکر
به چه روشی می‌توان یک vieamodel که ترکیبی از 3 model هست به مدلهای مربوطه به وسیله AutoMapper ، نگاشت کرد

نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۴/۰۳/۱۱ ۱:۴۳

از روش بیان شده در این لینک استفاده کنید

عنوان: **وی‌ژگی های پیشرفته ی AutoMapper - قسمت دوم**
نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۵/۰۲ ۱۳:۳۴
آدرس: www.dotnettips.info
گروه‌ها: AutoMapper

در ادامه قسمت قبلی به بررسی ویژگی‌های پیشرفته‌ی AutoMapper می‌پردازیم...

Custom type converters

همانطور که از اسمش مشخصه، زمانی کاربرد داره که نوع عضو یا اعضای یک شی در مبداء، با معادلشون در مقصد یکی نیستند.
مثلا فرض کنید نوع Bool در مبداء رو می‌خواهیم به نوع String در مقصد نگاشت کنیم؛ همون Yes و No معروف بجای True یا False .
کلاس‌های زیر رو در نظر بگیرید:

```
public class Source
{
    public string Value1 { get; set; }
    public string Value2 { get; set; }
    public string Value3 { get; set; }
}

public class Destination
{
    public int Value1 { get; set; }
    public DateTime Value2 { get; set; }
    public Type Value3 { get; set; }
}
```

طبق مستندات AutoMapper اگه بخواهیم این دو رو نگاشت کنیم Exception میده چون AutoMapper نمیدونه چطوری باید مثلا Int رو به String تبدیل کنه؛ برای همین ما باید به AutoMapper بگیم چطور این تبدیل نوع رو انجام بده.

نکته: در تستی که من انجام دادم، AutoMapper تبدیل نوع‌های ابتدایی رو خودش انجام میده؛ مثلا همین تبدیل Int به String رو!

یکی از روش‌های مهیا کردن تبدیل کننده‌ی نوع، پیاده سازی اینترفیس ITypeConverter<TSource, TDestination> هست. تقریبا مثل کاری که در WPF و SL با پیاده سازی اینترفیس IValueConverter انجام می‌دادیم.
من برای تست از همون تبدیل نوع Bool به String استفاده میکنم و البته بخاطر ساده بودن دیگه Model ها رو نمی‌نویسم.
ابتدا تعریف کلاس تبدیل کننده‌ی نوع:

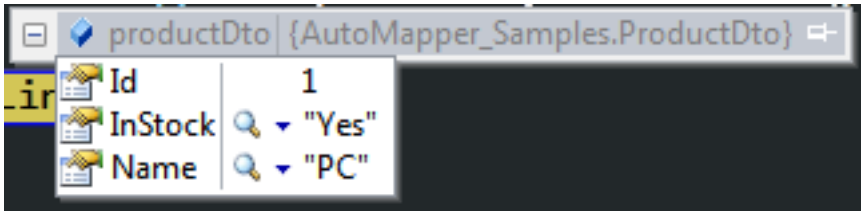
```
public class BooltoStringTypeConvertor : ITypeConverter<bool, string>
    {
        public string Convert(ResolutionContext context)
        {
            return (bool)context.SourceValue ? "Yes" : "No";
        }
    }
```

و نحوه استفاده:

```
Mapper.CreateMap<bool,string>().ConvertUsing<BooltoStringTypeConvertor>();
            Mapper.CreateMap<Product, ProductDto>();
            Mapper.AssertConfigurationIsValid();

var product = new Product { Id = 1,Name ="PC" ,InStock = true };
var productDto = Mapper.Map<Product, ProductDto>(product);
```

خروجی به شکل زیر میشه.



نکته: TypeConvertorها میدان دیدشون سراسریه و نیازی نیست به ازای هر نگاشتی اونو به AutoMapper معرفی کنیم Global Scope.

Custom value resolvers

کلاس‌های زیر رو در نظر بگیرید

```
public class Person
{
    public int Id { get; set; }

    public string FirstName { get; set; }

    public string LastName { get; set; }
}

public class PersonDTO
{
    public int Id { get; set; }

    public string RawData { get; set; }
}
```

فرض کنید داخل RawData تمامی اعضای شی مبداء رو به صورت Comma Delimited ذخیره کنیم. برای این کار می‌تونیم از Value Resolver استفاده کنیم.

یک روش برای این کار ارث بری از کلاس Abstract ی بنام ValueResolver<TSource, TDestination> هست.

```
public class CommaDelimetedResolver:ValueResolver<Person,string>
    {
        protected override string ResolveCore(Person source)
        {
            return string.Join(",", source.Id, source.FirstName, source.LastName);
        }
    }
```

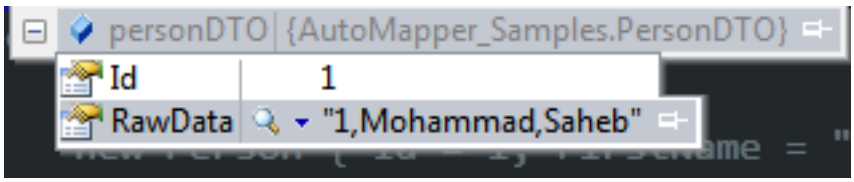
و نحوه استفاده

```
Mapper.CreateMap<Person, PersonDTO>().ForMember(
                des => des.RawData, op => op.ResolveUsing<CommaDelimetedResolver>());


var person = new Person
{
Id = 1,
FirstName = "Mohammad",
LastName = "Saheb",
};

var personDTO = Mapper.Map<Person, PersonDTO>(person);
```

و خروجی به شکل زیر میشه



نکته: توجه کنید این فقط یک مثال بود و این کار رو با روش‌های دیگه هم میشه انجام داد مثلا MapFrom و...
نکته: میدان دید Value Resolverها سراسری نیست و باید به ازای هر نگاشتی اونو معرفی کنیم.

Custom Value Formatters

فرض کنید تاریخ رو در بانک، به صورت میلادی ذخیره کرده‌اید و می‌خواهید سمت View به صورت شمسی نمایش بدید. بنابراین در مبدا ویژگی بنام MiladiDate از نوع DateTime دارید و در مقصد ویژگی بنام ShamsiDate از نوع String. هنگام نگاشت، AutoMapper به صورت پیش فرض ToString رو فراخونی میکنه که بدرد ما نمیخوره و...
برای این کار میشه از Value Formatter استفاده کرد با پیاده سازی اینترفیس IValueFormatter.

```
public class ShamsiFormatter:IValueFormatter
    {
        public string FormatValue(ResolutionContext context)
        {
            return ToShamsi(context.SourceValue.ToString());
        }
    }
```

نحوه استفاده

```
Mapper.CreateMap<Person, PersonDTO>().ForMember(
            des => des.ShamsiDate, op => op.AddFormatter<ShamsiFormatter>());
```

نظرات خوانندگان

نویسنده: torisoft
تاریخ: ۱۳۹۱/۰۵/۰۹ ۲:۱۴

سلام جناب صاحب
ببخشید من یه کاربر مبتدیم سوالم رو خیلی ساده میگم.
من یه پروژه سیلور mvvm دارم با سرویس web api
اول اینکه Automapper تو سیلور به چه صورت عمل میکنه ؟
دوم اینکه مدل‌های اصلی سمت سرور نوشته میشن. حالا DTO این مدل‌ها باید سمت سیلور نوشته بشن ؟
سوم اینکه امکان نگاشت دو یا چند مدل به صورت همزمان به یک DTO وجود دارد ؟
چهارم اینکه تعریف نگاشت باید در سمت سرور باشد یا سیلور ؟ به عبارت دیگه در کنترلر سمت سرور یا در viewmodel سمت سیلور یا هیچکدام ؟
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۹ ۸:۴۵

در سمت کلاینت سیلورلایت به صورت رسمی پشتیبانی نمی‌شود . سمت سرور آن هم همین مواردی است که تا الان گفته شده و زمانیکه دات نت فول در اختیار شما باشد مباحث آن یکی است و تفاوتی نمی‌کند.

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۵/۰۹ ۱۰:۳۸

با تشکر از آقای نصیری...
-3بله میشه که بهش Flattening میگن.

نویسنده: مسعود
تاریخ: ۱۳۹۱/۰۵/۱۰ ۲:۳۲

جناب نصیری و صاحب سلام
به چه صورت اطلاعات رو از context گرفته و map کنیم ؟
منظورم اینکه کد زیر بدون استفاده از automapper درسته

```
private TollContext db = new TollContext();

        private IQueryable<CarDTO> MapCars()
        {
            return from p in db.Cars select new CarDTO() { Id = p.Id, Name = p.Name, Price = p.Price };
        }

        public IEnumerable<CarDTO> GetCars()
        {
            return MapCars().AsEnumerable();
        }
```

ولی با استفاده از automapper (البته نمی‌دونم به جه صورته) به شکل زیر نوشتم که غلطه

```
private TollContext db = new TollContext();

        private IQueryable<CarDto> MapCars()
        {
            return Mapper.Map<Car, CarDto>(db.Cars);
        }

        public IEnumerable<CarDto> GetCars()
```

```
        {
            return MapCars().AsEnumerable();
        }
```

لطفا در صورت امکان راهنمائی بفرمائید.
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۰ ۸:۲۹

این پیشنیاز رو اول مطالعه کنید.

نویسنده: رضا بزرگی
تاریخ: ۱۳۹۱/۰۶/۰۲ ۱۹:۴۲

با سلام. لطفا اگر امکانش هست در مورد مپ کردن چند نوع داده‌ای مبدا به یک نوع داده‌ای مقصد هم توضیح دهید. ممنونم.
مثال:

```
public class SourceType1 //poco class
{
   public string Name
   public string JobTitle
   public string PetsName
}

public class SourceType2
{
  public string BankName
  public decimal CreditCardBalance
  public DateTime SignupDate
}

public class Destination
{
    public string Job
    public string Balance //thousand seprator
}
```

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۶/۰۳ ۱۴:۱۴

ممنون...
این مورد برا من پیش نیومده بود.
معمولا به این صورته که ما یک کلاس مرکب داریم و از اون میرسیم به کلاس سبک‌تر یا همون Flattening و البته یک روش برای حل مشکل شما هم همینه اون 2 تا کلاس مبدا رو تو یه کلاس مرکب داشته باشید و...
یا اینکه از این کلاس کمکی که اینجا معرفی شده استفاده کنید.
نحوه فراخونی به این شکل میشه

```
var Destination = EntityMapper.Map<Destination>(SourceType1, SourceType2);
```

نویسنده: میثم
تاریخ: ۱۳۹۱/۱۰/۱۵ ۲۰:۲۱

سلام ، مشکلی که من با Automapper دارم اینه
کلاسی به شکل زیر دارم

```
public class ProductType:BaseEntity
```

```
    {
        #region Field
        private string persianTitle;
        private string englishTitle;
        private IList<Product> products;
        #endregion
        #region Memebr
        public string PersianTitle
        {
            get
            {
                return (persianTitle);
            }
            set
            {
                persianTitle = Microsoft.Security.Application.Encoder.HtmlEncode(value);
            }
        }
        public string EnglishTitle
        {
            get
            {
                return (englishTitle);
            }
            set
            {
                englishTitle = Microsoft.Security.Application.Encoder.HtmlEncode(value);
            }
        }
        public virtual IList<Product> Products
        {
            get
            {
                return (products);
            }
            set
            {
                products = value;
            }
        }
        #endregion
    }
```

و خوب کلاسی با عنوان Product هم موجوده با کدی به این شکل سعی در آپدیت کردن این کلاس دارم

```
[HttpPost]
        public ActionResult Update(ProductTypeViewModel productTypeViewModel)
        {
            if (productTypeViewModel.ExaminId())
            {
                ProductType productType;
                productType = _productType.Find(x => x.Id == productTypeViewModel.Id);
                AutoMapper.Mapper.Map(productTypeViewModel, productType);
                _uow.SaveChanges();
            }
            return RedirectToAction("Index");
        }
```

متاسفانه خطا رخ میده ، خطایی مربوط به Context ظاهرا اینم پیام خطا

```
issing type map configuration or unsupported mapping.

Mapping types:
ProductTypeViewModel -> ProductType_5334DF7BAFBE780DF5328E2D6DF2A0DC3350F23340BE2EE2FC506AE9EDEC38DA
MvcUserInterface.Areas.Management.Models.ProductTypeViewModel ->
System.Data.Entity.DynamicProxies.ProductType_5334DF7BAFBE780DF5328E2D6DF2A0DC3350F23340BE2EE2FC506AE9E
DEC38DA

Destination path:
ProductType_5334DF7BAFBE780DF5328E2D6DF2A0DC3350F23340BE2EE2FC506AE9EDEC38DA

Source value:
MvcUserInterface.Areas.Management.Models.ProductTypeViewModel
```

ممنون میشم اگه کسی تجربه ای داره کمکم کنه من برای پیاده سازی لایه‌های سرویس ، دامین و دیتا از روشی که آقای نصیری گفته استفاده کردم . ویرایش کلاس هایی که دارای عضوی به صورت لیست از کلاس دیگر هستند چگونه باید باشه ؟ برای map کردن این کلاس‌ها کار خاصی باید انجام بشه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۵ ۲۱:۳

این نام کلاس‌های طولانی رو که مشاهده می‌کنید در حقیقت پشت صحنه EF است و کلاس‌های پروکسی نام دارند. بنابراین نیاز به کمی تنظیم بیشتر هست. ادامه در اینجا

نویسنده: میهمان
تاریخ: ۱۳۹۱/۱۰/۲۸ ۰:۵۲

آیا امکان تعریف AutoMapper در لایه‌های دیگر هم وجود دارد ؟ در صورت مثبت بودن پاسخ، چگونه اون رو توی لایه UI صدا بزنیم ؟

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۰/۲۸ ۸:۵۰

کار automapper نگاشت خواص لایه ui به domain برنامه است و برعکس. اینکار در mvc مثلا در کنترلرها انجام میشه. کنترلر هم نتیجه کار رو با return View به لایه نمایشی برای استفاده ارسال می‌کنه.

نویسنده: ali
تاریخ: ۱۳۹۱/۱۲/۰۱ ۱۱:۲۷

attributeهای مدل مانند Display را چرا وقتی Map میکنیم نمیاره؟

به طور مثال در صورتی میاره که به شکل زیر باشه

```
public class Customer
  {
    public Customer()
    {
      Orders = new List<Order>();
    }
    [StringLength(10)]
    public string Title { get; set; }

    [Display(Name = "نام")]
    public string FirstName { get; set; }

    [Display(Name = "نام خانوادگی")]
    public string LastName { get; set; }
    public ICollection<Order> Orders { get; set; }
}
```

```
public class CustomerViewModel
{
    public Customer Customer{ get; set; }
}
```

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۱۲/۰۱ ۱۴:۳۴

گویا این امکان موجود نیست من که خودم Data Annotationها رو تو ViewModel تکرار میکنم.
برای بررسی بیشتر لطفا کد قسمت کانفیگ Mapping و همچنین کلاسهاتون رو بنویسید.

نویسنده: ali
تاریخ: ۱۳۹۱/۱۲/۰۷ ۱۳:۹

map کرد میشه دو طرفه باشه.

به طور مثال :

```
Mapper.CreateMap<Customer, CustomerCreateVM>()
                .ForMember(f => f.Date, f => f.AddFormatter<PersianDateFormatter>());
```

و بعد واسه بر عکسش:

```
Mapper.CreateMap<CustomerCreateVM, Customer>()
                .ForMember(f => f.Date, f => f.AddFormatter<DateTimeFormatter>());
```

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۱۲/۰۷ ۱۴:۶

بله میتونید.
ولی گویا برای این کار یا Flatten و Unflatten کردن ;فریمورک valueinjecter توصیه میشه .

نویسنده: نوید
تاریخ: ۱۳۹۱/۱۲/۲۷ ۱۲:۱

ممنون از مطلب مفیدتون
من از کلاس کمکی ای که برای نگاشت چند کلاس به یک کلاس معرفی کردید استفاده کردم ولی به یک مشکل برخورد کرد.
من یک کلاس کالا دارم و یک کلاس برند که میخوام این دو رو به یک کلاس دیگه نگاشت بدم.
در اون کلاسی که معرفی کردید یک به یک کلاس‌های منبع را نگاشت میدهد. هنگام استفاده وقتی که که مثلا اول کلاس کالا و بعد کلاس برند رو برای نگاشت ارسال میکنم ، کلیه پارامتر هایی که از کلاس کالا در کلاس مقصد وجود دارند مقادیر صفر و Null میگیرند و فقط مقادیر کلاس برند در کلاس مقصد مقادیر درستشان را دارند. البته من در متد configure هنگام برای نگاشت کلاس دوم پارامترهای کلاس اول را Ignore کردم. ولی باز هم همین مشکل پیش اومد!
یک سوال دیگه هم داشتم :
بهتر نیست به جای اینکه ما در Controller دو تابع از هر کلاس را فراخوانی کنیم(یک تابع برای دریافت لیست کالاها و یکی برای لیست برندها) و کار نگاشت این دو به صورتیکه معرفی شد انجام دهیم، یک تابع که ترکیبی از دو کلاس را برمیگرداند (تابعی که با استفاده از Join کلاس‌ها مقادیر مورد نظرمان را برگرداند، یعنی برند هر کالا را به آن بچسباند) را اجرا کرده و سپس مقادیر آن را به صورت معمول نگاشت کنیم؟
با سپاس و ببخشید که طولانی شد

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۱۲/۲۷ ۱۲:۵۳

چک کن قبل مپ کردن پراپرتی هات نال نباشن.برای بررسی دقیقتر کد رو بزار

شدنش میشه و بستگی به طراحی شما تو لایه سرویس داره و در اون صورت دیگه نیازی به استفاده از این کلاس کمکی هم نداری.

نویسنده: نوید
تاریخ: ۱۳۹۱/۱۲/۲۷ ۱۶:۱۳

ممنون از پاسختون.
پراپرتی‌ها مقادیر درستی دارند و بعد از نگاشت مقادیر Null میگیرند.
این هم کدها :

```
public class Kala
```

```
{
        [Key]
        public int Kala_id { get; set; }

        [DisplayName("نام کالا")]
        public string Name { get; set; }

        [DisplayName("قیمت خرید")]
        public double Fee_Kharid { get; set; }

        public virtual Brand Brand { get; set; }
    }

  public class Brand
    {
        [Key]
        public int Brand_id { get; set; }
        public string Brand_Name { get; set; }
        public virtual ICollection<Kala> Kalas { get; set; }
    }

 public class KalaViewModel
    {
        public int Kala_Id { get; set; }
        public string  Name { get; set; }
        public double Fee_Kharid { get; set; }
        public string Brand_Name { get; set; }
    }

     //Controller
     [HttpGet]
        public ActionResult Index()
        {
            var kala = _Kala_Service.GetAllKalas();
            var brand = _Brand_Service.GetAllBrands();

            var kalaviewmodel = EntityMapper.Map<List<KalaViewModel>>(kala, brand);
            return View(kalaviewmodel);
        }

 protected override void Configure()
        {
            Mapper.CreateMap<Kala, KalaViewModel>();

            Mapper.CreateMap<Brand, KalaViewModel>()
                      .ForMember(des => des.Kala_Id, op => op.Ignore())
                      .ForMember(des => des.Name, op => op.Ignore())
                      .ForMember(x => x.Fee_Kharid, opt => opt.Ignore());

        }
```

سپاس

نویسنده: علیرضا پایدار
تاریخ: ۱۳۹۱/۱۲/۲۸ ۱۱:۲۱

```
public class Kala
{
        [Key]
        public int Kala_id { get; set; }
```

```
        [DisplayName("نام کالا")]
        public string Name { get; set; }

        [DisplayName("قیمت خرید")]
        public double Fee_Kharid { get; set; }

        public virtual Brand Brand { get; set; }
    }

 public class Brand
   {
       [Key]
       public int Brand_id { get; set; }
       public string Brand_Name { get; set; }
       public virtual ICollection<Kala> Kalas { get; set; }
   }

public class KalaViewModel
   {
       public int Kala_Id { get; set; }
       public string  Name { get; set; }
       public double Fee_Kharid { get; set; }
       public string Brand_Name { get; set; }
   }

    //Controller
    [HttpGet]
       public ActionResult Index()
       {
           var kala = _Kala_Service.GetAllKalas();
           var brand = _Brand_Service.GetAllBrands();

           var kalaviewmodel = EntityMapper.Map<List<KalaViewModel>>(kala, brand);
           return View(kalaviewmodel);
       }

protected override void Configure()
       {
           Mapper.CreateMap<Kala, KalaViewModel>()
           .ForMember(des => des. Brand_Name, op => op.MapFrom(src =>     src.Brand.Brand_Name) );
       }
```

کد پایین از لحاظ منطقی هم درست نیست

map کردن Brand با KalaViewModel معنایی نداره

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۱۲/۲۸ ۱۲:۱۰

تابع ignore باعث عدم مپ کردن اون پراپرتی میشه ولی همونطور که ذکر شده AutoMapper براساس قراردادها کار میکنه و این یه قرارداده که پراپرتی که در مبدا معادلی براش در مقصد نباشه به صورت دیفالت ignore میشه پس نیازی به ignore نیست.

```
Mapper.CreateMap<Kala, KalaViewModel>()
          .ForMember(des => des.Brand_Name, op => op.MapFrom(src => src.Brand.Brand_Name));

           var kalas = new[]
               {
                   new Kala
                       {
                           Kala_id = 1,
                           Brand = new Brand {Brand_id = 1, Brand_Name = "Nike"},
                           Fee_Kharid = 150000,
                           Name = "Shoes"
                       },new Kala
                       {
                           Kala_id = 2,
                           Brand = new Brand {Brand_id = 1, Brand_Name = "Nike"},
                           Fee_Kharid = 12000,
                           Name = "Shirt"
                       }
               };
```

```
var kalaviewmodel = Mapper.Map<Kala[], KalaViewModel[]>(kalas);
```

نویسنده: نوید
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۱:۱۸

سپاس

نویسنده: نوید
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۱:۵۹

ممنون از راهنماییتون .
یک سوال دیگه هم برام پیش اومد: اگه نحوه ارتباط کلاس‌ها به صورت زیر باشه:

```
public class Kala
    {
        [Key]
        public int Kala_id { get; set; }

        [DisplayName("نام کالا")]
        public string Name { get; set; }

        [DisplayName("قیمت خرید")]
        public double Fee_Kharid { get; set; }

        public virtual Brand Brand { get; set; }

        public ICollection<Anbar_Kala> Anbar_Kalas { get; set; }
    }

 public class Anbar_Kala
    {
        [ForeignKey("Anbar_Id")]
        public virtual Anbar Anbar { get; set; }
        public int Anbar_Id { get; set; }

        [ForeignKey("Kala_Id")]
        public virtual Kala Kala{ get; set; }
        public int Kala_Id { get; set; }

        [DisplayName("تعداد")]
        public int Tedad { get; set; } //تعداد کالاها در هر انبار
    }

public class KalaViewModel
    {
        public int Kala_Id { get; set; }
        public string  Name { get; set; }
        public double Fee_Kharid { get; set; }
        public string Brand_Name { get; set; }
        public int Tedad { get; set; }
    }

//controller
        var kala = _Kala_Service.GetAllKalas();
        var tedad= _Anbar_Kala_Service.GetAllAnbar_Kalas();

        var kalaviewmodel = EntityMapper.Map<List<KalaViewModel>>(kala, tedad);


protected override void Configure()
        {
        Mapper.CreateMap<Kala, KalaViewModel>()
       .ForMember(des => des.Brand_Name, op => op.MapFrom(src => src.Brand.Brand_Name));

       Mapper.CreateMap<Anbar_Kala, KalaViewModel>(); // این نگاشت باید به چه صورتی باشد؟
      .ForMember(des =>des.Kala_Id, op=>op.Ignore();
        }
```

تو شرایط فوق که نحوه ارتباط کلاس‌ها به صورت عکس حالت قبله، اگر به صورت بالا نگاشت صورت بگیره، قبل از نگاشت پراپرتی‌ها نال نیستند ولی بعد از نگاشت باز هم پارامترهای مربوط به کلاس کالا که در Modelview قرار دارند Null میشوند.

نویسنده: علیرضا پایدار
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۸:۵۷

دوست عزیز آوردن پراپرتی Tedad در کلاس KalaViewModel اشتباهه و همچنین map کردن Anbar_kala با آن.
کار زیر را میتونی انجام بدی:

```
public class Anbar_KalaViewModel
    {
        public Anbar Anbar { get; set; }
        public Kala Kala{ get; set; }
        public int Tedad { get; set; }
    }
//Class Configure
Mapper.CreateMap<Anbar_Kala,Anbar_KalaViewModel>();
```

نویسنده: حسن
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۹:۱۸

یک توصیه: مطلب ارزنده « اصول و قراردادهای نام‌گذاری در دات‌نت » بهتر است در کدهای شما رعایت شود.

نویسنده: مرتضی رییسی
تاریخ: ۱۳۹۴/۰۵/۰۱ ۱۳:۱۱

سلام.
استفاده از Ingore برای من اجباری شده. بدین صورت که مدل ویو من تعداد (نسبتا زیادی) پروپرتی نسبت به مدل برنامه ام، کم داره. حالا موقع تبدیل از مدل به ویو مدل مشکلی ندارم. ولی برعکس این موضوع خطای Unmapped members were found میده به همراه لیست این پروپرتی ها:

Unmapped properties:

Salt

LastLoginDateTime

BanedDateTime

LastUpdateDateTime

تعدادی از فیلدها رو Ignore کردم و از این لیست حذف شدند.
طبق گفته شما استفاده از Ignore اختیاریه ولی برای من اجباری شده. اگه این پروپرتی‌ها رو Ignore کنم خطایی بهم نخواهد داد.
میشه راهنمایی بفرمائید تا بصورت کلی بتونم از Ignore استفاده کنم و نخوام برا تک تک فیلدها اینکارو انجام بدم.
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۴/۰۵/۰۱ ۱۴:۷

« استفاده از Auto-ignore در AutoMapper »

عنوان: **نکته‌ای در استفاده از AutoMapper**
نویسنده: سیدمجتبی حسینی
تاریخ: ۱۳۹۱/۰۵/۲۱ ۲:۵۳
آدرس: www.dotnettips.info
گروه‌ها: MVC, AutoMapper

مدل Student را به شکل زیر در نظر بگیرید

```
public class Student
   {
       public int Id { get; set; }
       public string Name { get; set; }
       public string Family { get; set; }
       public DateTime Birthdate { get; set; }
       public string Tel { get; set; }
       public string CellPhone { get; set; }
       [Email]
       public string Email { get; set; }
   }
```

: آن را به صورت زیر ViewModel

```
public class StudentViewModel
    {
        public string Name { get; set; }
        public string Family { get; set; }
        public string Email { get; set; }
    }
```

برای نگاشت مجموعه‌ای از مدل Student به مجموعه‌ای از مدل StudentViewModel داریم:

```
public ActionResult Index()
   {
      var model = db.Students.ToList();
      AutoMapper.Mapper.CreateMap<Student,StudentViewModel>();
      var studentViewModel = AutoMapper.Mapper.Map<List<Student>,
IEnumerable<StudentViewModel>>(model);
      return View(studentViewModel);
    }
```

که اگر دستور

```
AutoMapper.Mapper.CreateMap<Student,StudentViewModel>();
```

را استفاده نکنیم، خطای زیر اتفاق می‌افتد:

```
An exception of type 'AutoMapper.AutoMapperMappingException' occurred in AutoMapper.dll but was not
handled in user code
```

نظرات خوانندگان

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۲۱ ۹:۲۶

سلام ،

در AutoMapper معرفی نگاشت‌ها از پیش شرط‌های انجام نگاشت اشیا به هم می‌باشد.
معمولا این نگاشت‌ها در ابتدای شروع برنامه (مثلا متد Application_Start در Global.asax) تعریف می‌شوند.

نویسنده: مجتبی حسینی
تاریخ: ۱۳۹۱/۰۵/۲۱ ۱۱:۵۷

با توجه به مطالب درج شده در این سایت، درباره AutoMapper و عدم ذکر نکته فوق خواستم این نکته را یادآوری کنم.
از تذکر خوب شما هم ممنونم

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۲۱ ۱۲:۱۰

خواهش می‌کنم.
فقط این مسئله رو بگم که در این پست اشاره شده که : "در شروع برنامه نگاشت‌ها رو تعریف میکنم.یک روش ابداعی تعریف نگاشت‌ها در یک کلاس استاتیک و فراخوانی اون تو متد Application_Start هست."

نویسنده: محسن
تاریخ: ۱۳۹۲/۱۰/۲۷ ۱۷:۱

با سلام به دوستان عزیزم
نحوه انتقال اتوماتیک validation attributeهای هر مدل به ویومدل مربوطه چگونه توسط automapper قابل انجامه؟ چون به صورت پیش فرض data annotationها مپ نمیشن، آیا تنظیم خاصی برای این کار مورد نیازه؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۷ ۱۷:۲۵

قبل از پرسش بهتره کمی جستجو کنید. مثلا یک مورد در اینجا بحث شده

نویسنده: ح مرادف
تاریخ: ۱۳۹۳/۱۱/۰۷ ۱۷:۷

سلام؛ در تمامی مثالها شما از مپر درون اکشن‌ها استفاده کردین، مگر لایه سرویس نباید خودش یک ویو-مدل بگیره و بقیه کارهای نگاشت و ... رو انجام بده و در خروجی نیز نوع ویو-مدل خروجی دهد ؟
من یکم گیج شدم. احساس می‌کنم عملیات اتومپر باید درون لایه سرویس انجام بشه که داخل اکشن فقط متدهای سرویس رو صدا بزنیم. از طرفی هم توی لایه سرویس که اومدم استفاده کنم که ویو-مدلم رو به کلاس ef نگاشت کنم و بعد بریزم توی دیتابیس دیدم که نمیشه ، چون لایه سرویس application_start برای کانفیگ اتومپر نداره ؟! لطفا راهنمایی نمایید. باتشکر

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۱۱/۰۷ ۱۸:۱۹

لایه سرویس با بقیه قسمت‌های برنامه درون یک app domain اجرا میشن. یعنی همینکه تنظیمات اولیه auto mapper رو در ابتدای برنامه انجام دادید، بقیه لایه‌ها هم می‌تونن ازش استفاده کنن. این تنظیمات singleton هستن. یعنی فقط یک وهله ازشون در کل

طول عمر برنامه وجود داره.

عنوان: **نگاشت دیتای XML به کمک AutoMapper**
نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۷/۱۴ ۱۳:۲۱
آدرس: www.dotnettips.info
برچسب‌ها: AutoMapper, LINQ to XML

صورت مساله که مشخصه قراره دیتای رو از منبع داده‌ی Xml به model مورد نظرمون نگاشت کنیم چیزی شبیه کاری که متد GetEntries انجام میده و تو این پست معرفی شده...

AutoMapper به صورت داخلی و با استفاده از قراردادها نمیتونه xml رو به object تبدیل کنه ولی این کار به کمک LINQ to XML قابل انجامه.

مثالی که برای این پست انتخاب شده سوژه‌ی داغ روزهای اخیره ؟!
مدل زیر رو در نظر داشته باشید

```
public class PreciousMetal
    {
        public string Name { get; set; }
        public float Price { get; set; }
        public DateTime UpdateTime { get; set; }
    }
```

قراره از یک وب سرویس اطلاعات مربوط به فلزات گرانبها رو دریافت و به مدل PreciousMetal نگاشت کنیم.ساختار اطلاعات دریافتی ما به شکل زیره

```
<pricelist currency="usd">
  <price timestamp="1349347920" per="ozt" commodity="gold">1788.70</price>
  <price timestamp="1349347860" per="ozt" commodity="palladium">665.50</price>
  <price timestamp="1349347920" per="ozt" commodity="platinum">1701.25</price>
  <price timestamp="1349347920" per="ozt" commodity="silver">34.91</price>
</pricelist>
```

برای نگاشت‌های معمولی کار سختی نداریم و از MapFrom استفاده میکنیم مثلا برای قیمت

```
Mapper.CreateMap<XElement, PreciousMetal>().ForMember(des => des.Price,
                                                       op =>
                                                       op.MapFrom(src => src.Value));
```

ولی برای زمان دریافت قیمت با توجه به متفاوت بودن زمان دریافتی مثلا در اینجا Unix time از Custom value resolvers استفاده میکنیم

```
public class UnixTimestampResolver : ValueResolver<XElement, DateTime>
    {
        protected override DateTime ResolveCore(XElement source)
        {
            var origin = new DateTime(1970, 1, 1, 0, 0, 0, 0);
            return origin.AddSeconds(Convert.ToDouble(source.Attribute("timestamp").Value));
        }
    }
```

همچنیا میخواهیم از معادل فارسی نام فلزات گرانبها استفاده کنیم

```
public class EnglishPMetalToFarsiResolver : ValueResolver<XElement, string>
    {
        readonly Dictionary<string, string> _pMetaldic = new Dictionary<string, string>
                                                         {
                                                             {"gold", "طلا"},
                                                             {"palladium", "پالادیوم"},
                                                             {"platinum", "پلاتین "},
                                                             {"silver", "نقره"}
```

```
                                                    };
        protected override string ResolveCore(XElement source)
        {
            string pMetalFarsi;
            return _pMetaldic.TryGetValue(source.Attribute("commodity").Value, out pMetalFarsi) ?
pMetalFarsi : string.Empty;
        }
    }
```

نکته: از سری قبلی آشنایی با AutoMapper همیشه بین انتخاب Custom Value Formatters و Custom value resolvers مشکل داشتم مثلا همین قسمت بنظر خودم Custom Value Formatters مناسبتر میاد بعد کمی وقت گذاشتن مشخص شد گویا یه جورایی Custom Value Formatters اضافه س و اشتباه تو طراحی بوده.

و اما نحوه استفاده

```
static void Main(string[] args)
        {
            //تعریف نگاشت‌ها Mapper.CreateMap<XElement, PreciousMetal>().ForMember(des => des.Name,
                                                                    op =>
op.ResolveUsing<EnglishPMetalToFarsiResolver>())
                .ForMember(des => des.Price,
                           op =>
                           op.MapFrom(src => src.Value))

                .ForMember(des => des.UpdateTime, op => op.ResolveUsing<UnixTimestampResolver>());

            Mapper.AssertConfigurationIsValid();

            //دریافت قیمت‌ها از منبع داده
            var doc = XDocument.Load("http://www.xmlcharts.com/cache/precious-metals.xml");
            var priceData = doc.Descendants("pricelist").Take(1).Elements("price");

            //فراخوانی نگاشت
            var preciousMetals = Mapper.Map<IEnumerable<XElement>, IList<PreciousMetal>>(priceData);


            foreach (var preciousMetal in preciousMetals)
            {
                Console.WriteLine(preciousMetal.Name + " " + preciousMetal.Price + " " +
preciousMetal.UpdateTime.ToShortDateString());
            }

            Console.ReadLine();

        }
```

عنوان: **چگونگی رسیدگی به Null property در AutoMapper**
نویسنده: مهراد
تاریخ: ۱۳۹۲/۰۲/۲۲ ۱۷:۴۰
آدرس: www.dotnettips.info
گروه‌ها: C#, MVC, Mapper, AutoMapper

AutoMapper کتابخانه‌ای برای نگاشت اطلاعات یک شیء به شیءایی دیگر به صورت خودکار می‌باشد.

در این مقاله چگونگی رسیدگی به Null property را در AutoMapper بررسی خواهیم کرد. فرض کنید شیء منبع دارای یک خاصیت Null است و می‌خواهید به وسیله Automaper شیء منبع را به مقصد نگاشت نمایید. اما می‌خواهید در صورت Null بودن شیء مبدا، یک مقدار پیش فرض برای شیء مقصد در نظر گرفته شود .

برای نمونه کلاس user را که در آن از کلاس Address یک خاصیت تعریف شده، در نظر بگیرید . اگر مقدار آدرس در شیء منبع خالی بود شاید شما بخواهید مقدار آن را به صورت empty string و یا با یک مقدار پیش فرض در مقصد مقدار دهی کنید.

همانند مثال زیر :

```
public class UserSource
{
  public Address Address{get;set;}
}

public class UserDestination
{
  public string Address{get;set;}
}
```

ابتدا نگاشت‌ها را تعریف می‌کنیم:

```
AutoMapper.Mapper.CreateMap<UserSource, UserDestination>()
          .ForMember(dest => dest.Address
          , opt => opt.NullSubstitute("Address not found")
          );
```

کد بالا نشان دهنده تبدیل Address به Address است ولی دارای متد اختیاری NullSubstitute می‌باشد و بیانگر این است که اگر آدرس شیء منبع Null بود، مقدار پیش فرضی را برای شیء مقصد در نظر بگیرد. در انتها می‌توان نگاشت را در برنامه متناسب با نیاز خود انجام داد:

```
var model = AutoMapper.Mapper.Map<UserSource, UserDestination>(user);
var models = AutoMapper.Mapper.Map<IEnumerable<UserSource>, IEnumerable<UserDestination>>(users);
```

نظرات خوانندگان

نویسنده: daneshjoo
تاریخ: ۱۳۹۲/۰۲/۲۳ ۲۰:۴۱

با سلام

مهندس من یه دیتابیس دارم که حاوی اطلاعات است در جداول اون در تمام ستون‌ها به غیر از ستون کلید اومده تیک alow null رو فعال کرده یعنی این ستون‌ها می‌تونه مقدار null رو بگیره .

حالا اون برنامه که این اطلاعات رو وارد دیتابیس کرده اومده هر ستونی که نوعش رشته بوده مقدار empty وارد کرده نه null و ستون هایی که نوعشون int هست مقدار صفر وارد کرده , مثل همین مطلبی که شما گفتید اما به صورت سنتی .

به نظر شما من باید همین رویه رو با روش شما انجام بدم یا نه همون مقدار null و در دیتابیس ذخیره کنم ؟

نویسنده: MehRad
تاریخ: ۱۳۹۲/۰۲/۲۳ ۲۱:۴۶

سلام
بستگی به کار خودتون داره . مطلب بالا مربوط به نگاشت اطلاعات یک شیء به شیءایی دیگره .اما شما برای در نظر گرفتن مقدار پیش فرض در دیتابیس همون طور که میدونید با تنظیم Default Value or Binding میتونی مقدار پیش فرضی برای ستون‌های Null در نظر بگیری.

نویسنده: ناصر پورعلی
تاریخ: ۱۳۹۳/۰۳/۲۱ ۱۶:۴۴

سلام
من از Automapper واسه مپ کردن مدل و ویو مادل‌ها تو برنامه ام استفاده میکنم
مشکلم اینجاست که در کلاس model یه فیلد از نوع byte[] دارم

```
public class News : BaseEntity
    {
public byte[] SmallImage { get; set; } // SmallImage
     }
```

در viewModel هم دقیقا همین فیلد رو دارم.
منتهی موقع مپ کردن این فیلد null میشه.

```
news = model.ToEntity(news);
```

```
public static News ToEntity(this NewsModel model, News destination)
        {
            Mapper.CreateMap<NewsModel, News>();
            return Mapper.Map(model, destination);
        }
```

علتش رو نمیدونم، کسی هست بدونه؟
باتشکر

نویسنده: علیرضا پایدار
تاریخ: ۱۳۹۳/۰۳/۲۲ ۹:۱۵

```
public class TestModel
    {
        public byte[] MProperty { get; set; }
    }
```

```
public class TestViewModel
    {
       public byte[] VMProperty { get; set; }
    }
```

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

Mapper.CreateMap<TestModel,TestViewModel>().ForMember(tv=>tv.VMProperty,m=>m.MapFrom(t=>t.MProperty));
            TestModel tm = new TestModel();
            tm.MProperty = new byte[]  { 1, 2, 3 ,4};

            TestViewModel tvm = Mapper.Map<TestModel, TestViewModel>(tm);
            foreach (var item in tvm.VMProperty)
            {
                Console.WriteLine(item.ToString());
            }
            Console.ReadKey();
        }
    }
```

در صورتی که MProperty، مقدار داشته باشه مشکلی پیش نمیاد اگر هم null باشه بازم مشکلی پیش نمیاد و VMProperty برابر با null هست.

عنوان: **خودکارسازی فرآیند نگاشت اشیاء در AutoMapper**
نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۴/۰۶/۰۱ ۱۷:۲۵
آدرس: www.dotnettips.info
گروه‌ها: Refactoring, AutoMapper

قرار دادن تمامی تنظیمات نگاشت‌ها درون کلاس‌های پروفایل تا حدودی حجم کدهای ما را در آینده زیاد خواهد کرد.

```
public class TestProfile1 : Profile
{
    protected override void Configure()
    {
        // این تنظیم سراسری هست و به تمام خواص زمانی اعمال می‌شود
        this.CreateMap<DateTime, string>().ConvertUsing(new DateTimeToPersianDateTimeConverter());
        this.CreateMap<User, UserViewModel>();
       // Other mappings
     }

    public override string ProfileName
    {
        get { return this.GetType().Name; }
    }
}
```

در ادامه می‌خواهیم به روشی جهت سازماندهی بهتر این نوع کلاس‌ها بپردازیم. به طوری‌که تعاریف مربوط به نگاشت‌ها در کنار View Modelهای برنامه قرار گیرند. برای اینکار ابتدا اینترفیس‌های زیر را ایجاد خواهیم کرد:

```
public interface IMapFrom<T>
{

}
public interface IHaveCustomMappings
{
     void CreateMappings(IConfiguration configuration);
}
```

خوب، همانطور که مشاهده می‌کنید، در اینترفیس IMapFrom امضای هیچ متدی تعریف نشده است. در واقع View Modelهای ما از این اینترفیس جهت تشخیص اینکه به چه مدلی قرار است نگاشت شوند، استفاده خواهند کرد. اما در حالتی‌که نیاز به نگاشت صریح پراپرتی‌های یک View Model داشتیم می‌توانیم اینترفیس IHaveCustomMappings را پیاده‌سازی کرده و جزئیات نگاشت را درون متد CreateMappings تعیین کنیم.
به عنوان مثال View Model زیر را در نظر بگیرید:

```
public class PersonViewModel : IMapFrom<Person>
{
      public string Name { get; set; }
      public string LastName { get; set; }
}
```

خوب، در اینجا با پیاده‌سازی اینترفیس IMapFrom نوع مبدا را برای ویومدل فوق مشخص کرده‌ایم. در این‌حالت هدف ما نگاشت تمامی خواص کلاس Person به تمامی خواص کلاس PersonViewModel خواهد بود. برای حالت‌های خاص نیز که نیاز به نگاشت دقیق خواص باشد به اینصورت عمل خواهیم کرد:

```
public class PersonViewModel : IHaveCustomMapping
{
     public string Name { get; set; }
     // دیگر پراپرتی‌ها

     public void CreateMappings(IConfiguration configuration)
     {
          configuration.CreateMap<ApplicationUser, PersonViewModel>()
               .ForMember(m => m.Name, opt =>
                    opt.MapFrom(u => u.ApplicationUser.UserName));
          // دیگر نگاشت‌ها
     }
}
```

خوب، در نهایت با استفاده از امکانات LINQ و Reflection کار پردازش تنظیمات نگاشت‌های هر View Model و خودکارسازی فرآیند نگاشت را انجام خواهیم داد. اینکار را می‌توانیم درون یک کلاس با نام AutoMapperConfig و با پیاده‌سازی اینترفیس IRunInit انجام دهیم:

```
public void Execute()
{
      var types = Assembly.GetExecutingAssembly().GetExportedTypes();

      LoadStandardMappings(types);

      LoadCustomMappings(types);
}
```

در داخل متد Execute دو متد به نام‌های LoadStandardMappings و LoadCustomMapping را فراخوانی کرده‌ایم. متد اول برای پردازش حالتی است که اینترفیس IMapFrom را پیاده‌سازی کرده باشیم و متد دوم نیز برای حالتی است که اینترفیس IHaveCustomMappings را پیاده‌سازی کرده باشیم.

متد LoadStandardMappings :

```
private static void LoadStandardMappings(IEnumerable <Type> types)
{
     var maps = (from t in types
                       from i in t.GetInterfaces()
                       where i.IsGenericType && i.GetGenericTypeDefinition() == typeof(IMapFrom< >)  &&
!t.IsAbstract && !t.IsInterface
                       select new {
                                Source = i.GetGenericArguments()[0],
                                Destination = t
                       }).ToArray();

      foreach(var map in maps)
      {
               Mapper.CreateMap(map.Source, map.Destination);
      }
}
```

توضیح کدهای فوق :

ابتدا تمامی typeهای تعریف شده در پروژه به متد فوق پاس داده خواهند شد.

برای هر type تمامی اینترفیس‌هایی که توسط این type پیاده‌سازی شده باشند را دریافت خواهیم کرد.

سپس هر type که اینترفیس IMapFrom را پیاده‌سازی کرده باشد را پردازش می‌کنیم.

سپس از نوع‌های Abstract و Interface صرفنظر خواهیم کرد.

انواع مبدا و مقصد را برای AutoMapper فراهم خواهیم کرد.

در نهایت AutoMapper براساس آنها نگاشت را ایجاد خواهد کرد.

متد LoadCustomMapping:

```
private static void LoadCustomMappings(IEnumerable <Type> types)
{
     var maps = (from t in types
                       from i in t.GetInterfaces()
                       where typeof(IHaveCustomMappings).IsAssignableFrom(t) && !t.IsAbstract &&
!t.IsInterface
                       select(IHaveCustomMappings) Activator.CreateInstance(t)).ToArray();

     foreach(var map in maps)
     {
               map.CreateMappings(Mapper.Configuration);
```

```
    }
}
```

توضیح کدهای فوق:

این متد نیز همانند متد قبلی، تمامی typeها را پردازش خواهد کرد. با این تفاوت که مواردی که اینترفیس IHaveCustomMappings را پیاده‌سازی کرده باشند، دریافت کرده و در نهایت متد CreateMappings آنها را فراخوانی خواهیم کرد. اکنون کدهای نگاشت برنامه از اصول Open and Closed پیروی می‌کنند. در نتیجه می‌توانیم نگاشت‌های جدید را به سادگی و با ایجاد View Model ها تعریف کنیم.