عنوان: **استفاده از StructureMap به عنوان یک IoC Container**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۲۶ ۱۴:۷
آدرس: www.dotnettips.info
گروه‌ها: Design patterns, Dependency Injection, IoC

StructureMap یکی از IoC containerهای بسیار غنی سورس باز نوشته شده برای دات نت فریم ورک است. امکان تنظیمات آن توسط کدنویسی و یا همان Fluent interfaces، به کمک فایل‌های کانفیگ XML و همچنین استفاده از ویژگی‌ها یا Attributes نیز میسر است. امکانات جانبی دیگری را نیز مانند یکی شدن با فریم ورک‌های Dynamic Proxy برای ساده سازی فرآیندهای برنامه نویسی جنبه‌گرا یا AOP، دارا است. در ادامه قصد داریم با نحوه استفاده از این فریم ورک IoC بیشتر آشنا شویم.

دریافت StructureMap

برای دریافت آن نیاز است دستور پاورشل ذیل را در کنسول نیوگت ویژوال استودیو فراخوانی کنید:

```
PM> Install-Package structuremap
```

البته باید دقت داشت که برای استفاده از StructureMap نیاز است به خواص پروژه مراجعه و سپس حالت Client profile را به Full profile تغییر داد تا برنامه قابل کامپایل باشد (در برنامه‌های دسکتاپ البته)؛ از این جهت که StructureMap ارجاعی را به اسمبلی استاندارد System.Web دارد.

آشنایی با ساختار برنامه

ابتدا یک برنامه کنسول را آغاز کرده و سپس یک Class library جدید را به نام Services نیز به آن اضافه کنید. در ادامه کلاس‌ها و اینترفیس‌های زیر را به Class library ایجاد شده، اضافه کنید. سپس از طریق نیوگت به روشی که گفته شد، StructureMap را به پروژه اصلی (ونه پروژه Class library) اضافه نمائید و Target framework آن‌را نیز در حالت Full قرار دهید بجای حالت Client profile.

```
namespace DI03.Services
{
    public interface IUsersService
    {
        string GetUserEmail(int userId);
    }
}

namespace DI03.Services
{
    public interface IEmailsService
    {
        void SendEmailToUser(int userId, string subject, string body);
    }
}

using System;

namespace DI03.Services
{
    public class UsersService : IUsersService
    {
        public UsersService()
        {
            //هدف صرفا نمایش وهله سازی خودکار این وابستگی است
            Console.WriteLine("UsersService ctor.");
        }

        public string GetUserEmail(int userId)
        {
            //برای مثال دریافت از بانک اطلاعاتی و بازگشت یک نمونه جهت آزمایش برنامه
            return "name@site.com";
        }
    }
```

```
}

using System;

namespace DI03.Services
{
    public class EmailsService: IEmailsService
    {
        private readonly IUsersService _usersService;
        public EmailsService(IUsersService usersService)
        {
            Console.WriteLine("EmailsService ctor.");
            _usersService = usersService;
        }

        public void SendEmailToUser(int userId, string subject, string body)
        {
            var email = _usersService.GetUserEmail(userId);
            Console.WriteLine("SendEmailTo({0})", email);
        }
    }
}
```

در لایه سرویس برنامه، یک سرویس کاربران و یک سرویس ارسال ایمیل تدارک دیده شده‌اند.
سرویس کاربران بر اساس آی دی یک کاربر، برای مثال از بانک اطلاعاتی ایمیل او را بازگشت می‌دهد. سرویس ارسال ایمیل، نیاز به ایمیل کاربری برای ارسال ایمیلی به او دارد. بنابراین وابستگی مورد نیاز خود را از طریق تزریق وابستگی‌ها در سازنده کلاس و وهله سازی شده در خارج از آن (معکوس سازی کنترل)، دریافت می‌کند.
در سازنده‌های هر دو کلاس سرویس نیز از Console.WriteLine استفاده شده‌است تا زمان وهله سازی خودکار آن‌ها را بتوان بهتر مشاهده کرد.
نکته مهمی که در اینجا وجود دارد، بی‌خبری لایه سرویس از وجود IoC Container مورد استفاده است.

استفاده از لایه سرویس و تزریق وابستگی‌ها به کمک StructureMap

```
using DI03.Services;
using StructureMap;

namespace DI03
{
    class Program
    {
        static void Main(string[] args)
        {
            // تنظیمات اولیه برنامه که فقط یکبار باید در طول عمر برنامه انجام شود
            ObjectFactory.Initialize(x =>
            {
                x.For<IEmailsService>().Use<EmailsService>();
                x.For<IUsersService>().Use<UsersService>();
            });

            //نمونه‌ای از نحوه استفاده از تزریق وابستگی‌های خودکار
            var emailsService = ObjectFactory.GetInstance<IEmailsService>();
            emailsService.SendEmailToUser(userId: 1, subject: "Test", body: "Hello!");
        }
    }
}
```

کدهای برنامه را به نحو فوق تغییر دهید. در ابتدا نحوه سیم کشی‌های آغازین برنامه را مشاهده می‌کنید. برای مثال کدهای ObjectFactory.Initialize باید در متدهای آغازین یک پروژه قرار گیرند و تنها یکبار هم نیاز است فراخوانی شوند.
به این ترتیب IoC Container ما زمانیکه قرار است object graph مربوط به IEmailsService درخواستی را تشکیل دهد، خواهد دانست ابتدا به سازنده‌ی کلاس EmailsService می‌رسد. در اینجا برای وهله سازی این کلاس به صورت خودکار، باید وابستگی‌های آن‌را نیز وهله سازی کند. بنابراین بر اساس تنظیمات آغازین برنامه می‌داند که باید از کلاس UsersService برای تزریق خودکار وابستگی‌ها در سازنده کلاس ارسال ایمیل استفاده نماید.
در این حالت اگر برنامه را اجرا کنیم، به خروجی زیر خواهیم رسید:

```
UsersService ctor.
EmailsService ctor.
SendEmailTo(name@site.com)
```

بنابراین در اینجا با مفهوم Object graph نیز آشنا شدیم. فقط کافی است وابستگی‌ها را در سازنده‌های کلاس‌ها تعریف کرده و سیم کشی‌های آغازین صحیحی را نیز در ابتدای برنامه معرفی نمائیم. کار وهله سازی چندین سطح با تمام وابستگی‌های متناظر با آن‌ها در اینجا به صورت خودکار انجام خواهد شد و نهایتا یک شیء قابل استفاده بازگشت داده می‌شود.
ابتدایی‌ترین مزیت استفاده از تزریق وابستگی‌ها امکان تعویض آن‌ها است؛ خصوصا در حین Unit testing. اگر کلاسی برای مثال قرار است با شبکه کار کند، می‌توان پیاده سازی آن‌را با یک نمونه اصطلاحا Fake جایگزین کرد و در این نمونه تنها نتیجه‌ی کار را بازگشت داد. کلاس‌های لایه سرویس ما تنها با اینترفیس‌ها کار می‌کنند. این تنظیمات قابل تغییر اولیه IoC container مورد استفاده هستند که مشخص می‌کنند چه کلاس‌هایی باید در سازنده‌های کلاس‌ها تزریق شوند.

تعیین طول عمر اشیاء در StructureMap

برای اینکه بتوان طول عمر اشیاء را بهتر توضیح داد، کلاس سرویس کاربران را به نحو زیر تغییر دهید:

```
using System;

namespace DI03.Services
{
    public class UsersService : IUsersService
    {
        private int _i;
        public UsersService()
        {
            //هدف صرفا نمایش وهله سازی خودکار این وابستگی است
            Console.WriteLine("UsersService ctor.");
        }

        public string GetUserEmail(int userId)
        {
            _i++;
            Console.WriteLine("i:{0}", _i);
            //برای مثال دریافت از بانک اطلاعاتی و بازگشت یک نمونه جهت آزمایش برنامه
            return "name@site.com";
        }
    }
}
```

به عبارتی می‌خواهیم بدانیم این کلاس چه زمانی وهله سازی مجدد می‌شود. آیا در حالت فراخوانی ذیل،

```
//نمونه‌ای از نحوه استفاده از تزریق وابستگی‌های خودکار
var emailsService1 = ObjectFactory.GetInstance<IEmailsService>();
emailsService1.SendEmailToUser(userId: 1, subject: "Test1", body: "Hello!");

var emailsService2 = ObjectFactory.GetInstance<IEmailsService>();
emailsService2.SendEmailToUser(userId: 1, subject: "Test2", body: "Hello!");
```

ما شاهد چاپ عدد 2 خواهیم بود یا عدد یک:

```
UsersService ctor.
EmailsService ctor.
i:1
SendEmailTo(name@site.com)
UsersService ctor.
EmailsService ctor.
i:1
SendEmailTo(name@site.com)
```

همانطور که ملاحظه می‌کنید، به ازای هربار فراخوانی ObjectFactory.GetInstance، یک وهله جدید ایجاد شده است. بنابراین مقدار i در هر دو بار مساوی عدد یک است.

اگر به هر دلیلی نیاز بود تا این رویه تغییر کند، می‌توان بر روی طول عمر اشیاء تشکیل شده نیز تاثیر گذار بود. برای مثال تنظیمات آغازین برنامه را به نحو ذیل تغییر دهید:

```
// تنظیمات اولیه برنامه که فقط یکبار باید در طول عمر برنامه انجام شود
ObjectFactory.Initialize(x =>
{
   x.For<IEmailsService>().Use<EmailsService>();
   x.For<IUsersService>().Singleton().Use<UsersService>();
});
```

اینبار اگر برنامه را اجرا کنیم، به خروجی ذیل خواهیم رسید:

```
UsersService ctor.
EmailsService ctor.
i:1
SendEmailTo(name@site.com)
EmailsService ctor.
i:2
SendEmailTo(name@site.com)
```

بله. با Singleton معرفی کردن تنظیمات UsersService، تنها یک وهله از این کلاس ایجاد خواهد شد و نهایتا در فراخوانی دوم ObjectFactory.GetInstance، شاهد عدد i مساوی 2 خواهیم بود (چون از یک وهله استفاده شده است).

حالت‌های دیگر تعیین طول عمر مطابق متدهای زیر هستند:

```
Singleton()
HttpContextScoped()
HybridHttpOrThreadLocalScoped()
```

با انتخاب حالت HttpContext، به ازای هر HttpContext ایجاد شده، کلاس معرفی شده یکبار وهله سازی می‌گردد.
در حالت ThreadLocal، به ازای هر Thread، وهله‌ای متفاوت در اختیار مصرف کننده قرار می‌گیرد.
حالت Hybrid ترکیبی است از حالت‌های HttpContext و ThreadLocal. اگر برنامه وب بود، از HttpContext استفاده خواهد کرد در غیراینصورت به ThreadLocal سوئیچ می‌کند.

شاید بپرسید که کاربرد مثلا HttpContextScoped در کجا است؟
در یک برنامه وب نیاز است تا یک وهله از DbContext (مثلا Entity framework) را در اختیار کلاس‌های مختلف لایه سرویس قرار داد. به این ترتیب چون هربار new Context صورت نمی‌گیرد، هربار هم اتصال جداگانه‌ای به بانک اطلاعاتی باز نخواهد شد. نتیجه آن رسیدن به یک برنامه سریع، با سربار کم و همچنین کار کردن در یک تراکنش واحد است. چون هربار فراخوانی new Context به معنای ایجاد یک تراکنش جدید است.
همچنین در این برنامه وب قصد نداریم از حالت طول عمر Singleton استفاده کنیم، چون در این حالت یک وهله از Context در اختیار تمام کاربران سایت قرار خواهد گرفت (و DbContext به صورت Thread safe طراحی نشده است). نیاز است به ازای هر کاربر و به ازای طول عمر هر درخواست، تنها یکبار این وهله سازی صورت گیرد. بنابراین در این حالت استفاده از HttpContextScoped توصیه می‌شود. به این ترتیب در طول عمر کوتاه Object graphهای تشکیل شده، فقط یک وهله از DbContext ایجاد و استفاده خواهد شد که بسیار مقرون به صرفه است.
مزیت دیگر مشخص سازی طول عمر به نحو HttpContextScoped، امکان Dispose خودکار آن به صورت زیر است:

```
protected void Application_EndRequest(object sender, EventArgs e)
{
  ObjectFactory.ReleaseAndDisposeAllHttpScopedObjects();
}
```

تنظیمات خودکار اولیه در StructureMap

اگر نام اینترفیس‌های شما فقط یک I در ابتدا بیشتر از نام کلاس‌های متناظر با آن‌ها دارد، مثلا مانند ITest و کلاس Test هستند؛

فقط کافی است از قراردادهای پیش فرض StructureMap برای اسکن یک یا چند اسمبلی استفاده کنیم:

```
// تنظیمات اولیه برنامه که فقط یکبار باید در طول عمر برنامه انجام شود
ObjectFactory.Initialize(x =>
{
   //x.For<IEmailsService>().Use<EmailsService>();
   //x.For<IUsersService>().Singleton().Use<UsersService>();
   x.Scan(scan =>
   {
       scan.AssemblyContainingType<IEmailsService>();
       scan.WithDefaultConventions();
   });
});
```

در این حالت دیگر نیازی نیست به ازای اینترفیس‌های مختلف و کلاس‌های مرتبط با آن‌ها، تنظیمات اضافه‌تری را تدارک دید. کار یافتن و برقراری اتصالات لازم در اینجا خودکار خواهد بود.

دریافت مثال قسمت جاری

DI03.zip

به روز شده‌ی این مثال‌ها را بر اساس آخرین تغییرات وابستگی‌های آن‌ها از مخزن کد ذیل می‌توانید دریافت کنید:

Dependency-Injection-Samples

نظرات خوانندگان

نویسنده: فرشید علی اکبری
تاریخ: ۱۳۹۲/۰۱/۲۷ ۱۱:۱۱

سلام
بسیار عالی ... مفاهیم اساسی و پایه ای برای درک بهتر استفاده از StrructureMap و مخصوصا قسمت scan آن برای من خیلی جالب بود.

نویسنده: فرشید علی اکبری
تاریخ: ۱۳۹۲/۰۱/۲۷ ۱۳:۴۲

مجددا خسته نباشید

مهندس جان کد زیر رو نگاه کنین :

```
ObjectFactory.Configure(c =>
{
    c.For<IUnitOfWork>().CacheBy(InstanceScope.Hybrid).Use<My_Context>();
    c.For<ICityService>().Use<EFCityService>();
    c.For<ICountryService>().Use<EFCountryService>();
});
```

درحالیکه Resharper میگه که CachBy منسوخ شده و باید از LifecycleIs استفاده کنی میخوام بدونم چطوری باید تغییرش بدم که حالت InstanceScope.Hybrid هم براش تعیین شده باشه؟ درضمن اگه رفرنس خوبی برای تسلط به کار با StructureMap درنظر دارین رو محبت کنین ولینکش رو بذارین ممنون میشم.
با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۲۷ ۱۳:۴۸

- ماخذ خوب، مستندات رسمی آن است.
- توضیح دادم در متن. از متد HybridHttpOrThreadLocalScoped استفاده کنید. تمام این حالت‌ها خلاصه شدن به سه متد زیر:
(حالت هیبرید، بسته به نوع ویندوزی یا وب بودن برنامه به صورت خودکار نوع بهینه رو انتخاب می‌کنه)

```
Singleton()
HttpContextScoped()
HybridHttpOrThreadLocalScoped()
//و مثال
x.For<IUsersService>().HybridHttpOrThreadLocalScoped().Use<UsersService>();
```

نویسنده: رضا بزرگی
تاریخ: ۱۳۹۲/۰۲/۰۷ ۲۰:۳۰

تعیین طول عمر اشیاء در حالت تنظیمات اولیه خودکار به چه صورت است؟ آیا در این روش میشود برای هر کلاس طول عمرهای متفاوتی تعریف کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۷ ۲۱:۲۹

به این ترتیب:

```
scan.ConnectImplementationsToTypesClosing(typeof(ITestService))
      .OnAddedPluginTypes(y => y.HybridHttpOrThreadLocalScoped());
```

نویسنده: فرشید علی اکبری
تاریخ: ۱۳۹۲/۰۲/۱۸ ۹:۵۲

من تا روز قبل هیچ مشکلی دراستفاده از StructureMap نداشتم وازکار کردن باهاش لذت می‌بردم ولی با استفاده از NuGet دیروز آخرین نسخه StructureMap رو از سایت گرفتم وموقع اجرای برنامه به محض اولین وهله سازی (uow) اشکال میگیره و پیغام Error in the application میده.... ولی اگه دستی و بدون استفاده از StructureMap وهله سازی کنم کارها روی روال پیش میره.... Error Code =205 و توی سایت‌ها زیاد سرچ زدم ولی موارد قید شده راهگشای این مشکل نبود...درضمن توی Error Details هم پیغام زیر رو نمایش میده :

```
StructureMap Exception Code:  205
Missing requested Instance property "path" for InstanceKey "fcfc9943-37d0-45d3-aebc-dad30fe29e59"
```

که اگه ConnectionString من مشکل داره پس چرا بدون استفاده از StructureMap اون وهله سازی میکنه؟
و کدهای ایجاد و وهله سازی :

```
ObjectFactory.Initialize(x =>
                {

x.For<IUnitOfWorkCentralSystem>().HybridHttpOrThreadLocalScoped().Use<ContextCentralSystem>();
                    x.For<IYearService>().Use<YearService>();
                    x.For<IUserService>().Use<UsersService>();
                });
            var _uow = ObjectFactory.GetInstance<IUnitOfWorkCentralSystem>();
```

پیشاپیش ممنون از همکاری شما.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱۰:۱۷

آیا کلاس ContextCentralSystem دارای سازنده‌ای است با پارامتر path که باید مقدار دهی شود؟ اگر بله روش کار شبیه به کد زیر است:

```
x.For<IUnitOfWorkCentralSystem>()
 .HybridHttpOrThreadLocalScoped()
 .Use<ContextCentralSystem>()
 .Ctor<string>("path").Is("........."); //مقدار دهی سازنده با پارامتر رشته‌ای خاص
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱۱:۷

ضمن اینکه روش دیگری نیز برای بازگشت دادن یک وهله، وجود دارد:

```
x.For<IUnitOfWork>().HybridHttpOrThreadLocalScoped().Use(() =>
{
    var ctx = new Sample07Context();
    ctx.Database.Connection.ConnectionString = "...";
    return ctx;
});
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۲۲ ۱۳:۴۰

مقایسه‌ای بین توانمندی‌های کتابخانه‌های Ioc Container مختلف در دات نت . ماخذ

نویسنده: سیدعلی
تاریخ: ۱۳۹۲/۰۷/۰۱ ۱۱:۱۷

این هم یک مقایسه دیگر که فکر می‌کنم جمع بندی آن به عهده توسعه دهندگان و انتخاب آنها بستگی دارد.

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۲/۰۷/۰۱ ۲۲:۴۵

آیا جهت خودکار سازی تنظیمات اولیه باید ابتدا یک نمونه از اینترفیس هایمان را به StructureMap معرفی کنیم تا به صورت خوکار کار اسکن اسمبلی هایمان را انجام دهد؟
مثلاً در مثال شما جهت اسکن اسمبلی‌ها :

```
x.Scan(scan =>
   {
       scan.AssemblyContainingType<IEmailsService>();
       scan.WithDefaultConventions();
   });
```

به عنوان مثال شما IEmailService را تعریف کرده اید.
درست متوجه شدم؟
و سوال دیگر آیا امکان مشخص کردن Namespace جهت اسکن اسمبلی‌ها توسط StructureMap وجود دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۱ ۲۳:۵۴

- IEmailsService (یا کلا هر نوع مشخصی) می‌تواند در یک اسمبلی دلخواه جداگانه باشد. ذکر آن کار اسکن را سریعتر و دقیق‌تر می‌کند. فقط یک نوع علامتگذاری است؛ این اسمبلی خاص رو بگرد، نه جای دیگری را.
- بله. برای نمونه باید IAssemblyScanner را پیاده سازی کنید. اطلاعات بیشتر
و یا یک مثال در اینجا StructureMap - Don't Scan All Assemblies

نویسنده: سیدعلی
تاریخ: ۱۳۹۲/۰۸/۲۸ ۱۹:۱۵

سلام
من در یک پروژه که دارای دو قسمت ویندوز سرویس و وب می‌باشد از تزریق وابستگی با **StructureMap** استفاده می‌کنم که دارای دو تعریف از UOW برای این دو هستم می‌خواستم بدونم به چه شکلی آن را تعریف کنم که در هر زمان که لازم بود یک نمونه از آن را با توجه به پروژه ای که هستم (ویندوز سرویس یا وب) داشته باشم و آیا اصولا این کار امکان پذیر است؟

```
x.For<UnitOfWorkScopeBase<TModelUnitOfWork>>()
                    .HttpContextScoped()
                    .Use<PerThreadUnitOfWorkScope<TModelUnitOfWork>>();
x.For<UnitOfWorkScopeBase<TModelUnitOfWork>>()
                   .HybridHttpOrThreadLocalScoped()
                   .Use<PerThreadUnitOfWorkScope<TUnitOfWork>>();
```

با تشکر از شما

نویسنده: وحید نصیری

تاریخ: ۱۳۹۲/۰۸/۲۸ ۱۹:۲۵

- در متن بحث شده: « حالت Hybrid ترکیبی است از حالت‌های HttpContext و ThreadLocal. اگر برنامه وب بود، از HttpContext استفاده خواهد کرد در غیراینصورت به ThreadLocal سوئیچ می‌کند.»
یعنی HybridHttpOrThreadLocalScoped هر دو مورد را به صورت خودکار پوشش می‌دهد.
- ضمنا روش تشخیص کلی زمینه برنامه جاری، اینکه وب است یا ویندوز به صورت زیر است:

```
using System.Web;

        bool IsInWeb
        {
            get
            {
                return HttpContext.Current != null;
            }
        }
```

این مورد ارجاعی را به اسمبلی استاندارد System.Web نیاز دارد.

نویسنده: سیدعلی
تاریخ: ۱۳۹۲/۰۸/۲۸ ۱۹:۴۸

با تشکر از پاسخگویتان بله این مطالب را می‌دانستم شاید منظورم را اشتباه رساندم اینکه از کدام دو حالت UOW خودم استفاده کنم به کدام شکل تعریف می‌شود یعنی در زمانی که از HybridHttpOrThreadLocalScoped استفاده می‌کنم PerThreadUnitOfWorkScope یا PerThreadUnitOfWorkScope باشد؟و خودش این دو را با توجه به وب یا ویندوز سرویس بودن تشخیص دهد. با توجه به اینکه این دو Context را به عنوان ورودی دریافت می‌کنند. که اگر دریافت نمی‌کردن نحوه تعریف آنها شاید به این شکل صحیح بود.

```
x.For<UnitOfWorkScopeBase<TrackingModelUnitOfWork>>()
                    .HybridHttpOrThreadLocalScoped()
                    .Use(() =>
                    {
                        IsWeb ?new PerRequestUnitOfWorkScope<TrackingModelUnitOfWork>(context) : new
PerThreadUnitOfWorkScope<TrackingModelUnitOfWork>(context)
                    });
```

در بالای آن Context را تعریف کردم اما نمی‌دانم مورد بالا را به چه نحوی تعریف کنم.

```
x.For<LightSpeedContext<TUnitOfWork>>().Singleton().Use(() =>
                {
                    var ctx = new LightSpeedContext<TrackingModelUnitOfWork>()
                    {
                        ConnectionString = "MyConnection",
                        QuoteIdentifiers = true,
                        DataProvider = DataProvider.SqlServer2012
                    };
                    return ctx;
                });
```

ممنون از راهنمایتان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۸ ۲۰:۵۲

- نیازی به PerThreadUnitOfWorkScope و PerRequestUnitOfWorkScope کتابخانه‌های ثالث در حین کار با StructureMap نیست. خود این IoC Container قابلیت مدیریت طول عمر اشیاء را دارد. HttpContextScoped آن یعنی مدیریت طول عمر یک شیء و زنده نگه داشتن آن در طول یک درخواست یا Request. بنابراین نیازی نیست یکبار StructureMap این‌کار را انجام دهد و یکبار دیگر هم کتابخانه‌ی ثالث دیگری که مثلا PerRequestUnitOfWorkScope در آن تعریف شده؛ کار اضافی است. (بحث «تعیین طول

عمر اشیاء در StructureMap» در متن فوق)
- فقط از HybridHttpOrThreadLocalScoped استفاده کنید تا هر دو حالت برنامه‌های وب و ویندوز را با یک تنظیم پوشش دهید. نیازی هم به بررسی IsInWeb یاد شده نیست. خود StructureMap به صورت توکار این کار را انجام می‌دهد.
- نیازی نیست تا کار وهله سازی را در قسمت Use انجام دهید (کار اضافی است). فقط نام کلاس آن‌را ذکر کنید کافی است.

```
x.For<IMyInterface>().HybridHttpOrThreadLocalScoped().Use<MyClass>();
```

در این حالت کلاس MyClass، هر سازنده‌ای داشته باشد، با توجه به سایر x.For-Use های نوشته شده به صورت خودکار سازنده‌های آن‌ها توسط StructureMap وهله سازی می‌شوند. تا n سطح هم باشد، کار وهله سازی آن‌ها خودکار است به شرطی که در تنظیمات StructureMap ذکر کنید، هر تزریق اینترفیس صورت گرفته در سازنده کلاسی با چه کلاسی مرتبط است یعنی x.For-Use های کاملی باید داشته باشید.

نویسنده: رضا گرمارودی
تاریخ: ۱۳۹۲/۰۹/۲۵ ۸:۵۱

سلام؛ برای اعمال توکار دات نت چه کار باید کرد. مثلا زمانی که CustomRole یا CustomMemberShip داریم و متد سازنده ما کانتکس و کلاس‌های دیتالایر را به عنوان پارامتر ورودی میگیره ، این گونه موارد و نمی‌تونم تزریق وابستگی‌ها را انجام داد. من اشتباه می‌کنم یا راه دیگه ای داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۲۵ ۹:۳

در قسمت بررسی الگوی Service locator توضیح داده شده‌است. جایی که نمی‌توانید کار وهله سازی اشیاء را مستقیما تحت کنترل قرار دهید، نیاز است از Service locator استفاده کنید. در حین کار با StructureMap اگر متد ObjectFactory.GetInstance مستقیما داخل کدهای کلاس بکارگرفته شود، مفهوم Service locator را دارد.

نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۰۹/۲۶ ۱۳:۳۹

من از structure در پروژه م به صورتی که توضیح دادین استفاده کردم.
در یه مورد خاص null هست. وقتی نیاز به پارشال اکشنی دارم که در کنترل دیگری قرار داره، درست کار میکنه سیم کشی‌ها و هیچ چیزی نال نیست.، اما وقتی نیاز دارم که پارشالی از اکشن کنترل جاری که در حال رندر هست ، استفاده کنم، نال هست همه‌ی اینترفیس ها. سازنده کنترلر هم فراخونی نمیشه.
ساختار کنترلر به این صورت هست:

```
public partial class ContactController : Controller
    {
        private IGroupsBusiness _groupsBusiness;
        private IContactsBusiness _contactsBusiness;

        public ContactController(IContactsBusiness contactsBusiness, IGroupsBusiness groupsBusiness)
        {
            _groupsBusiness = groupsBusiness;
            _contactsBusiness = contactsBusiness;
        }



        public virtual ActionResult View(int id)
        {
            var model = _contactsBusiness.Select(id);
            return View(model);
        }

        public virtual ActionResult ViewGroups(int contactId)
        {
            var model = _groupsBusiness.SelectByContactId(contactId);
```

```
            return PartialView(model);
        }
}
```

ابتدا view اجرا میشه و سیم کشی برقرار هست. داخل ویو ارجاعی به اکشن viewgroups داره. اما این بار نال هست و به مشکل برمیخورم.
من توی ویو نوشتم

```
@{ Html.RenderAction(MVC.Contact.ViewGroups(Model.Id)); }
```

اگر این اکشن رو بذارم داخل کنترلر دیگه و صداش بزنم کار میکنه.
آیا نباید کد بالا درست کار بکنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۲۶ ۱۴:۳۷

جهت رعایت بهتر نظم در سایت:
- هر دوره در سایت، یک قسمت مخصوص پرسش و پاسخ‌های شخصی مرتبط با آن دوره دارد.
- دوره جاری یک قسمت مجزای MVC دارد.
- نحوه ارسال یک گزارش خطای خوب را هم یکبار مطالعه کنید. ارسال stack trace و اصل خطای حاصل خیلی مهم است و بدون آن پاسخ دادن از راه دور، بسیار مشکل.

نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۳:۴

تشکر
مشکل حل شد.
خطای منطقی بود. یه اکشن به اسم View دارم در کنترلر. و بعد وقتی در اکشن‌های دیگه‌ی این کنترلر، return view رو صدا میزدم که اطلاعات نمایش داده بشه، این متد رو صدا میزده و بعد یه حلقه بی پایان و در انتها خطای نامعلوم از طرف structuremap صادر میشد

نویسنده: vici
تاریخ: ۱۳۹۲/۱۱/۰۶ ۲۲:۱۰

سلام؛ وقتی برنامه اجرا میشه مسیر برنامه به این صورته؟
ابتدا به یه وهله از اینترفیس Email می‌سازه وبعد متد SendEmailToUser صدا زده میشه ، داخل این متد متوجه میشه که به وهله ایی از کلاس user نیاز داره پس به صورت خودکار وهله سازی رو انجام میده ، مقدار لازمه برگشت داده میشه و بقیه عملیات درسته؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۰۶ ۲۲:۱۷

بله. البته از اینترفیس وهله سازی نمی‌شود. بر اساس تنظیمات ObjectFactory.Initialize ، می‌داند که درخواست رسیده به IEmailsService باید به کمک کلاس EmailsService وهله سازی شود و همینطور الی آخر.

نویسنده: داریوش حمیدی
تاریخ: ۱۳۹۳/۰۷/۱۳ ۱۵:۱۱

سلام؛ زیر ObjectFactoryیه خط سبز میاد و می‌نویسه :

```
Warning
```

```
Structuremap.ObjectFactory is obsolete objectfactory will be removed future 4.0 release of
structuremapfavor the usage of the container class for futur work
```

چطور می‌تونم برطرف کنم. ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۱۳ ۱۵:۵۷

در انجمن آن بیشتر بحث شده؛ در اینجا
در نگارش بعدی، ObjectFactory استاتیک حذف می‌شود. بجای آن باید بنویسید:

```
var container = new Container(x => {
// تنظیمات در اینجا
});
```

و بعد مثلا:

```
var controller = container.GetInstance(controllerType) as SomeType;
```

در جاهائیکه مستقیما با ObjectFactory کار می‌کردید، بهتر است IContainer آن‌را مورد استفاده قرار دهید:

```
public class MyController
{
    public MyController(IContainer container)
    {
    }
}
```

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۷/۲۷ ۲۱:۵۹

سلام؛ توی این ورژن جدید کدها رو به صورت زیر تغییر دادم. آیا درسته؟ ممنون میشم بررسی بفرمایید

```
public static class IoC
    {
        public static IContainer Initialize()
        {
            var container = new Container(x =>
            {

               x.For<IUnitOfWork>().HybridHttpOrThreadLocalScoped().Use(() => new baranDbContext());
                x.For<IUserService>().Use<UserService>();
                x.For<IUserMetaDataService>().Use<UserMetaDataService>();
            });

            return container;
        }
        public class StructureMapControllerFactory : DefaultControllerFactory
        {
            private readonly IContainer _container;

            public StructureMapControllerFactory(IContainer container)
            {
                _container = container;
            }
            protected override IController GetControllerInstance(RequestContext requestContext, Type
controllerType)
            {
                if (controllerType == null)
                    return null;

                return (IController)_container.GetInstance(controllerType);
```

```
            }
        }
}
```

و یه سول دیگه اینکه در یک فایل جدا معادل دستور زیر چی میشه؟ چون از این خط خطا میگیره

```
var userService = ObjectFactory.GetInstance<IUserService>();
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۲۸ ۱:۲۹

هر زمانیکه ObjectFactory حذف شد، آن‌را با پیاده سازی زیر جایگزین کنید. کار کردن با آن هم از طریق ObjectFactory.Container خواهد بود.

```
public static class ObjectFactory
    {
        private static readonly Lazy<Container> _containerBuilder =
            new Lazy<Container>(defaultContainer, LazyThreadSafetyMode.ExecutionAndPublication);

        public static IContainer Container
        {
            get { return _containerBuilder.Value; }
        }

        private static Container defaultContainer()
        {
            return new Container(x =>
            {
                // تنظیمات در اینجا
            });
        }
    }
```