عنوان: **تغییر PartCreationPolicy پیش فرض در MEF**
نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۲/۰۳/۲۲ ۸:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: Default, MEF, Dependency Injection, PartCreationPolicy

**تشریح مسئله :** در MEF به صورت پیش فرض نوع نمونه ساخته شده از اشیا به صورت Singleton است. در صورتی که بخواهیم یک نمونه جدید از اشیا به ازای هر درخواست ساخته شود باید PartCreationPolicyAttribute رو به ازای هر کلاس مجددا تعریف کنیم و نوع اون رو به NonShared تغییر دهیم. در پروژه‌های بزرگ این مسئله کمی آزار دهنده است. برای تغییر رفتار Container در MEF هنگام نمونه سازی Objectها باید چه کار کرد؟

**نکته:** آشنایی با مفاهیم MEF برای درک بهتر مطالب الزامی است.

*در صورتی که با مفاهیم MEF آشنایی ندارید می‌توانید از اینجا شروع کنید.

در MEF سه نوع PartCreationPolicy وجود دارد:
#1 Shared : آبجکت مورد نظر فقط یک بار در کل طول عمر Composition Container ساخته می‌شود.(Singleton)
#2 NonShared : آبجکت مورد نظر به ازای هر درخواست دوباره نمونه سازی می‌شود.
#3 Any : از حالت پیش فرض CompositionContainar برای نمونه سازی استفاده میشود که همان مورد اول است(Shared)

در اکثر پروژه‌ها ساخت نمونه اشیا به صورت Singleton میسر نیست و باعث اشکال در پروژه می‌شود. برای حل این مشکل باید PartCreationPolicy رو برای هر شی مجزا تعریف کنیم. برای مثال

```
[Export]
    [PartCreationPolicy( CreationPolicy.NonShared )]
    internal class ShellViewModel : ViewModel<IShellView>
    {
        private readonly DelegateCommand exitCommand;


        [ImportingConstructor]
        public ShellViewModel( IShellView view )
            : base( view )
        {
            exitCommand = new DelegateCommand( Close );
        }
    }
```

حال فرض کنید تعداد آبجکت شما در یک پروژه بیش از چند صد تا باشد. در صورتی که یک مورد را فراموش کرده باشید و UnitTest قوی و مناسب در پروژه تعبیه نشده باشد قطعا در طی پروژه مشکلاتی به وجود خواهد آمد و امکان Debug سخت خواهد شد.
برای حل این مسئله بهتر است که رفتار Composition Container رو در هنگام نمونه سازی تغییر دهیم. یعنی آبجکت‌ها به صورت پیش فرض به صورت NonShared تولید شوند و در صورت نیاز به نمونه Shared این Attribute رو در کلاس مورد نظر استفاده کنیم. کافیست از کلاس Composition Container که قلب MEF محسوب می‌شود ارث برده و رفتار مورد نظر را Override کنیم.
برای نمونه :

```
public class CustomCompositionContainer : CompositionContainer
{
    public CustomCompositionContainer(ComposablePartCatalog catalog)
        : base(catalog)
    {
    }

    protected override IEnumerable<Export> GetExportsCore(ImportDefinition definition)
    {
        definition = AdaptDefinition(definition);

        return base.GetExportsCore(definition);
    }
```

```
    private ImportDefinition AdaptDefinition(ImportDefinition definition)
    {
        ContractBasedImportDefinition namedDefinition = definition as ContractBasedImportDefinition;
        if (namedDefinition != null && namedDefinition.RequiredCreationPolicy == CreationPolicy.Any)
        {
            definition = new ContractBasedImportDefinition(namedDefinition.ContractName,
                                                           namedDefinition.RequiredMetadata,
                                                           namedDefinition.Cardinality,
                                                           namedDefinition.IsRecomposable,
                                                           namedDefinition.IsPrerequisite,
                                                         CreationPolicy.NonShared);
        }

        return definition;
    }
}
```

مشاهده می‌کنید که متد GetExportCore در کلاس بالا Override شده است و توسط متد AdaptDefinition اگر PartCreationPolicy به صورت Any بود نمونه ساخته شده به صورت NonShared ایجاد می‌شود.

حال فقط کافیست در پروژه به جای استفاده از CompositionContainer از CustomCompositionContainer استفاده کنیم.