عنوان: **MEF و الگوی Singleton**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۴/۱۴ ۱۹:۱۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Design patterns

در مورد معرفی مقدماتی MEF می‌توانید به این مطلب مراجعه کنید و در مورد الگوی Singleton به اینجا .

کاربردهای الگوی Singleton عموما به شرح زیر هستند:
1) فراهم آوردن دسترسی ساده و عمومی به DAL (لایه دسترسی به داده‌ها)
2) دسترسی عمومی به امکانات ثبت وقایع سیستم در برنامه logging -
3) دسترسی عمومی به تنظیمات برنامه
و موارد مشابهی از این دست به صورتیکه تنها یک روش دسترسی به این اطلاعات وجود داشته باشد و تنها یک وهله از این شیء در حافظه قرار گیرد.

با استفاده از امکانات MEF دیگر نیازی به نوشتن کدهای ویژه تولید کلاس‌های Singleton نمی‌باشد زیرا این چارچوب کاری دو نوع روش وهله سازی از اشیاء (PartCreationPolicy) را پشتیبانی می‌کند: Shared و NonShared . حالت Shared دقیقا همان نام دیگر الگوی Singleton است. البته لازم به ذکر است که حالت Shared ، حالت پیش فرض تولید وهله‌ها بوده و نیازی به ذکر صریح آن همانند ویژگی زیر نیست:

```
[PartCreationPolicy(CreationPolicy.Shared)]
```

مثال:
فرض کنید قرار است از کلاس زیر تنها یک وهله بین صفحات یک برنامه‌ی Silverlight توزیع شود. با استفاده از ویژگی Export به MEF اعلام کرده‌ایم که قرار است سرویسی را ارائه دهیم :

```
using System;
using System.ComponentModel.Composition;

namespace SlMefTest
{
   [Export]
   public class WebServiceData
   {
       public int Result { set; get; }

       public WebServiceData()
       {
           var rnd = new Random();
           Result = rnd.Next();
       }
   }

}
```

اکنون برای اثبات اینکه تنها یک وهله از این کلاس در اختیار صفحات مختلف قرار خواهد گرفت، یک User control جدید را به همراه یک دکمه که مقدار Result را نمایش می‌دهد به برنامه اضافه خواهیم کرد. دکمه‌ی دیگری را نیز به همین منظور به صفحه‌ی اصلی برنامه اضافه می‌کنیم.
کدهای صفحه اصلی برنامه (که از یک دکمه و یک Stack panel جهت نمایش محتوای یوزر کنترل تشکیل شده) به شرح بعد هستند:

```
<UserControl x:Class="SlMefTest.MainPage"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
```

```
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
   xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
   mc:Ignorable="d" d:DesignHeight="300" d:DesignWidth="400">
   <StackPanel>
       <Button Content="MainPageButton" Height="23"
               HorizontalAlignment="Left"
               Margin="10,10,0,0" Name="button1"
               VerticalAlignment="Top" Width="98" Click="button1_Click" />
       <StackPanel Name="panel1" Margin="5"/>
   </StackPanel>
</UserControl>
```

```
using System.ComponentModel.Composition;
using System.Windows;

namespace SlMefTest
{
   public partial class MainPage
   {
       [Import]
       public WebServiceData Data { set; get; }

       public MainPage()
       {
           InitializeComponent();
           this.Loaded += mainPageLoaded;
       }

       void mainPageLoaded(object sender, RoutedEventArgs e)
       {
           CompositionInitializer.SatisfyImports(this);
           panel1.Children.Add(new SilverlightControl1());
       }

       private void button1_Click(object sender, RoutedEventArgs e)
       {
           MessageBox.Show(Data.Result.ToString());
       }
   }
}
```

با استفاده از ویژگی Import به MEF اعلام می‌کنیم که به اطلاعاتی از نوع شیء WebServiceData نیاز داریم و توسط متد CompositionInitializer.SatisfyImports کار وهله سازی و پیوند زدن export و import های همانند صورت می‌گیرد. سپس استفاده‌ی مستقیم از Data.Result مجاز بوده و مقدار آن null نخواهد بود.

کدهای User control ساده اضافه شده به شرح زیر هستند:

```
<UserControl x:Class="SlMefTest.SilverlightControl1"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
   xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
   mc:Ignorable="d"
   d:DesignHeight="300" d:DesignWidth="400">

   <Grid x:Name="LayoutRoot" Background="White">
       <Button Content="UserControlButton"
               Height="23"
               HorizontalAlignment="Left"
               Margin="10,10,0,0"
               Name="button1"
               VerticalAlignment="Top"
               Width="125"
               Click="button1_Click" />
   </Grid>
</UserControl>
```

```
using System.ComponentModel.Composition;
```

```
using System.Windows;

namespace SlMefTest
{
   public partial class SilverlightControl1
   {
       [Import]
       public WebServiceData Data { set; get; }

       public SilverlightControl1()
       {
           InitializeComponent();
           this.Loaded += silverlightControl1Loaded;
       }

       void silverlightControl1Loaded(object sender, RoutedEventArgs e)
       {
           CompositionInitializer.SatisfyImports(this);
       }

       private void button1_Click(object sender, RoutedEventArgs e)
       {
           MessageBox.Show(Data.Result.ToString());
       }
   }
}
```

اکنون قبل از شروع برنامه یک break point را در سازنده‌ی کلاس WebServiceData قرار دهید. سپس برنامه را آغاز نمائید. تنها یکبار این سازنده فراخوانی خواهد شد (هر چند در دو کلاس کار Import اطلاعات WebServiceData صورت گرفته است). همچنین با کلیک بر روی دو دکمه‌ای که اکنون در صفحه‌ی اصلی برنامه ظاهر می‌شوند، فقط یک عدد مشابه نمایش داده می‌شود (با توجه به اینکه اطلاعات هر دکمه در یک وهله‌ی جداگانه قرار دارد؛ یکی متعلق است به صفحه‌ی اصلی و دیگری متعلق است به user control اضافه شده).