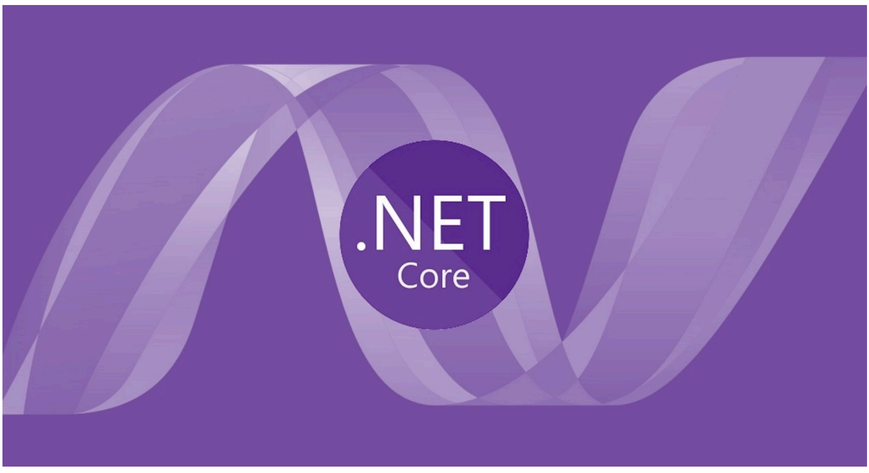
Prism：打造WPF项目的MVVM之选，简化开发流程、提高可维护性

作者：微信公众号：【架构师老卢】 11-28 15:34



概述：探索WPF开发新境界，借助Prism MVVM库，实现模块化、可维护的项目。强大的命令系统、松耦合通信、内置导航，让您的开发更高效、更流畅

在WPF开发中，一个优秀的MVVM库是`Prism`。以下是`Prism`的优点以及基本应用示例：

优点：

1. **模块化设计：** `Prism`支持模块化开发，使项目更易维护和扩展。
2. **强大的命令系统：** 提供了`DelegateCommand`等强大的命令实现，简化了用户交互操作的绑定。
3. **松耦合的通信：** 通过`EventAggregator`实现松耦合的组件间通信，提高了代码的可维护性。
4. **内置导航系统：** 提供了灵活的导航框架，支持导航到不同的视图和传递参数。

使用步骤：

1. 安装Prism NuGet包

在项目中执行以下命令：

```
Install-Package Prism.Wpf
```

2. 创建ViewModel

```
using Prism.Mvvm;

public class MainViewModel : BindableBase
{
    private string _message;

    public string Message
    {
        get { return _message; }
        set { SetProperty(ref _message, value); }
    }
}
```

3. 创建View

```
<Window x:Class="YourNamespace.MainWindow"
        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"
        xmlns:prism="http://prismlibrary.com/"
        prism:ViewModelLocator.AutoWireViewModel="True"
        mc:Ignorable="d"
        Title="MainWindow" Height="350" Width="525">
    <Grid>
        <TextBlock Text="{Binding Message}" />
    </Grid>
</Window>
```

4. 注册ViewModel

在`App.xaml.cs`中注册ViewModel：

```
using Prism.Ioc;
using Prism.Unity;
using YourNamespace.Views;

namespace YourNamespace
{
    public partial class App : PrismApplication
    {
        protected override Window CreateShell()
        {
            return Container.Resolve<MainWindow>();
        }

        protected override void RegisterTypes(IContainerRegistry containerRegistry)
        {
            containerRegistry.RegisterForNavigation<YourView>();
        }
    }
}
```

5. 在View中使用ViewModel

```
<Grid>
    <TextBlock Text="{Binding Message}" />
    <Button Command="{Binding UpdateMessageCommand}" Content="Update Message" />
</Grid>
```

6. 在ViewModel中处理命令

```
using Prism.Commands;

public class MainViewModel : BindableBase
{
    private string _message;

    public string Message
    {
        get { return _message; }
        set { SetProperty(ref _message, value); }
    }

    public DelegateCommand UpdateMessageCommand { get; }

    public MainViewModel()
    {
        UpdateMessageCommand = new DelegateCommand(UpdateMessage);
    }

    private void UpdateMessage()
    {
        Message = "Hello, Prism!";
    }
```

```
    }
```

以上是使用Prism的基本示例。Prism提供了更多的功能，如模块化开发、事件聚合器、导航框架等，以帮助构建结构良好、可维护的WPF应用。