MVVM Light Toolkit

Julita Borkowska

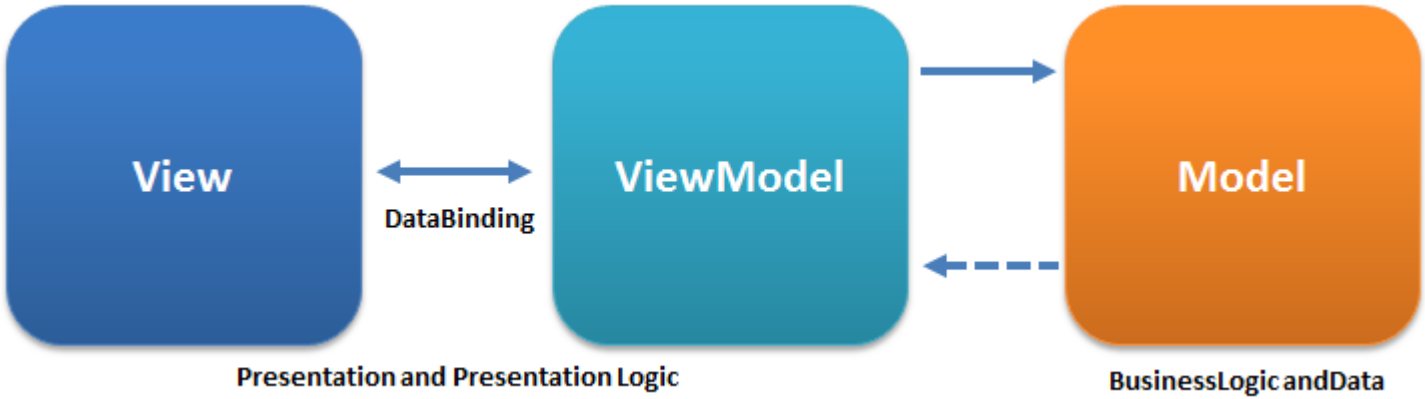
Czym jest MVVM Light Toolkit?

MVVM Light Toolkit został stworzony w 2009 roku przez Laurenta Bugnion.

Jest to biblioteka dostarczająca zestaw komponentów pomocnych podczas pisania aplikacji według wzorca MVVM.

Jej głównym celem jest przyśpieszenie tworzenia aplikacji o takiej architekturze oraz ich rozwój w projektach WPF, SilverLight, Windows Store i Windows Phone.

Wzorzec MVVM



Przy zastosowaniu MVVM zachodzi potrzeba przestrzegania następujących założeń:

- unikania kodu w code-behind – w większości przypadków to, co kiedyś było robione w code-behind, można przenieść do ViewModel,
- zdarzenia powinny zostać zastąpione komendami, np. zamiast podpinać zdarzenie Click, należy skorzystać z komendy; oczywiście istnieją przypadki, w których zdarzenia są jedynym rozwiązaniem,
- ViewModel powinien implementować interfejs INotifyPropertyChanged,
- dane z widoku powinny być powiązane z właściwościami w ViewModel,
- w testach sam ViewModel powinien wystarczyć; widok jest tak naprawdę wizualizacją przeznaczoną dla użytkownika; użytkownik, chcąc skorzystać z logiki dostarczonej przez aplikację, wprowadza tekst np. za pomocą TextBox – w testach jednostkowych ustawiamy właściwość w VM i powinniśmy uzyskać taki sam efekt,
- należy rozróżnić Model od ViewModel; model nie może zawierać żadnej logiki, związanej z widokiem; innymi słowy, model to czysta logika biznesowa, z kolei ViewModel zawiera już informacje o stanie widoku.

Co daje nam wykorzystanie MVVM Light Toolkit?

RelayCommand - lokalna implementacja interfejsu *ICommand*; w MVVM Light Toolkit odpowiada za reprezentowanie komend.

Dzięki użyciu *RelayCommand* nie musimy tworzyć skomplikowanego kodu, aby powiązać komendę z wywołaniem wskazanej metody.

Możemy tworzyć różne akcje, które później zostaną podpięte do widoku:

-klasa ta w konstruktorze oczekuje obiektu typu *Action* (czyli delegacji, która nie przyjmuje parametrów oraz nic nie zwraca), który będzie odpowiadał funkcji *Execute()* z interfejsu *ICommand*. Metoda ta jest wykonywana w momencie wywołania komendy.

Można również podać drugi argument będący obiektem typu *Predicate* (delegacja nie przyjmująca parametrów i zwracająca bool), który będzie implementował metodę *CanExecute()* - określa ona czy spełnione są warunki, aby móc wykonać daną komendę.

Istnieje również wersja generyczna obiektu *RelayCommand* - pozwala ona na przekazywanie parametru do komendy.

EventToCommand

W kontrolkach WPF’a komendy podpinane są domyślnie do jednego z góry przewidzianego zdarzenia, np. dla przycisku jest to odpowiednik onClick. Nie ma możliwości podłączenia komendy do innych zdarzeń. Tutaj własnie przychodzi z pomocą MVVM Light Toolkit i ***EventToCommand***.

Dzięki tej funkcjonalności, istnieje możliwość podłączenia komend z obiektu ViewModel do dowolnych zdarzeń kontrolki bez angażowania do tego jakiegokolwiek Code Behind – wszystko pozostaje więc w zgodzie z założeniami MVVM.

SimpleIoC - prosty kontener IoC, który pozwala nam na automatyczne tworzenie instancji wybranych obiektów

ViewModelBase - bazowa klasa dla naszych ViewModeli, implementująca interfejs INotifyPropertyChanged (interfejs ten ma za zadanie informować klienta, że wartość obiektu uległa zmianie)

Messenger

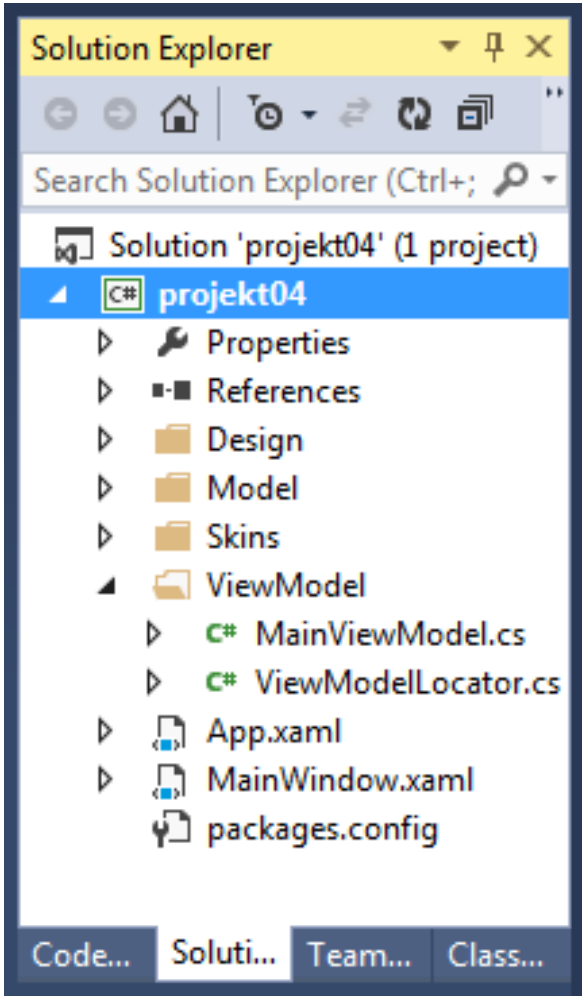
Z założenia obiekt Messenger został dodany do MVVM Light Toolkit jako narzędzie do wszechstronnej komunikacji - nie tylko pomiędzy obiektami ViewModel, ale pomiędzy dowolnymi klasami.

Idea opiera się na statycznym obiekcie Messenger, który udostępnia mechanizmy do **wysyłania komunikatów** oraz **rejestrowania akcji** reagujących na konkretne komunikaty.

Dodatkowo MVVM Light Toolkit zapewnia nam elementy mające przyspieszyć i ułatwić tworzenie kodu:

- **snippety** - gotowe fragmenty kodu,
- **szablony projektów i elementów**.

Wygląd projektu



App.xaml

```
<Application x:Class="projekt05.App"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:vm="clr-namespace:projekt05.ViewModel"
             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
             StartupUri="MainWindow.xaml"
             mc:Ignorable="d">

    <Application.Resources>
        <!--Global View Model Locator-->
        <vm:ViewModelLocator x:Key="Locator"
                             d:IsDataSource="True" />
    </Application.Resources>

</Application>
```

ViewModelLocator.cs

```
public class ViewModelLocator
{
    static ViewModelLocator()
    {
        ServiceLocator.SetLocatorProvider(() => SimpleIoc.Default);
        SimpleIoc.Default.Register<MainViewModel>();
    }

    public MainViewModel Main
    {
        get
        {
            return ServiceLocator.Current.GetInstance<MainViewModel>();
        }
    }
```

Dziękuję za uwagę.