Model-View-ViewModel (MVVM)

Design Patterns

Introduction

- Where did MVVM come from?
- What is MVVM?
- Components of MVVM
- Implementation

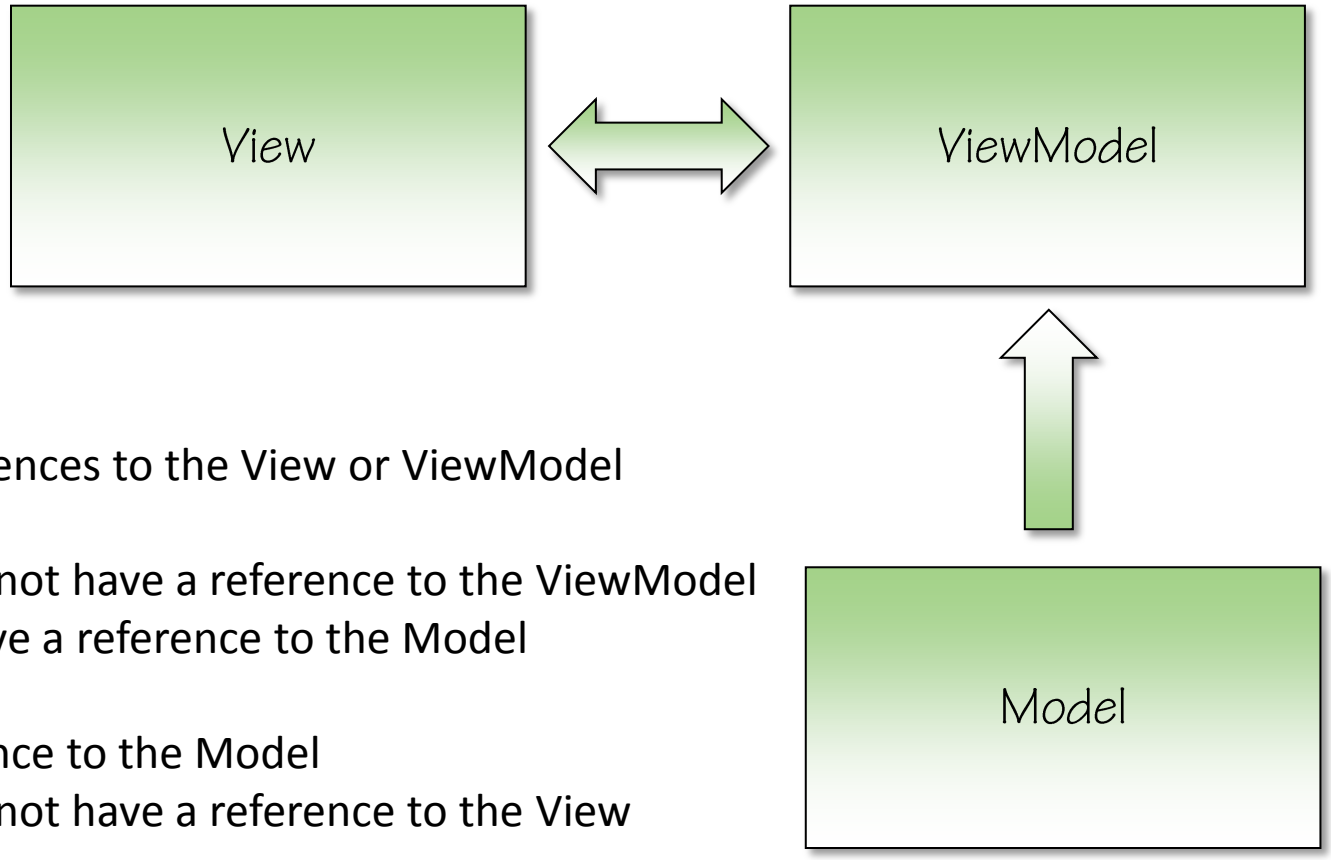
History

- **2004 Martin Fowler – Presentation Model (PM)**
  - Separates a view from it's state and behavior
  - Not dependent on a specific UI framework
- **2005 John Gossman unveiled the MVVM pattern**
  - Variation of MVC pattern
- **2008 John changes his mind**
  - Identical to PM pattern
  - Dependent on WPF/Silverlight

Intent

- **Separate concerns**
  - View
  - View's state and behavior
  - Data
- **Unit Testing & UI Testing**
- **Maintenance**
- **Extensibility**
- **Enables the designer/developer workflow**
- **Take advantage of WPF/Silverlight data binding**

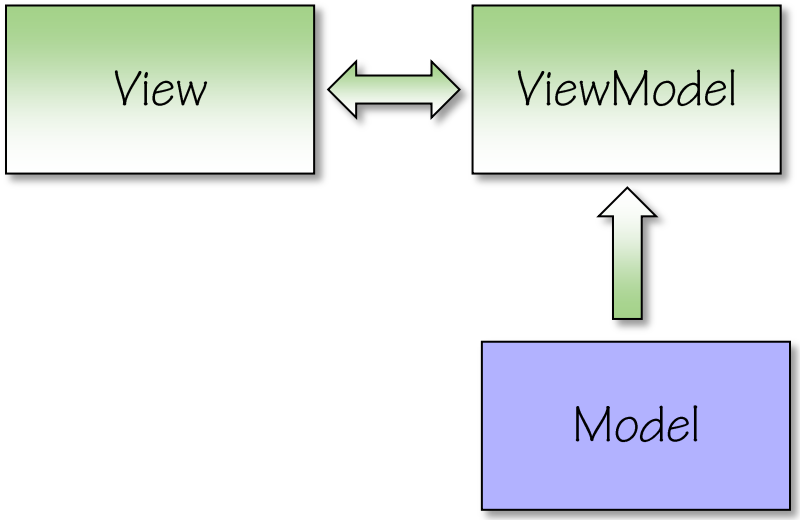
Structure



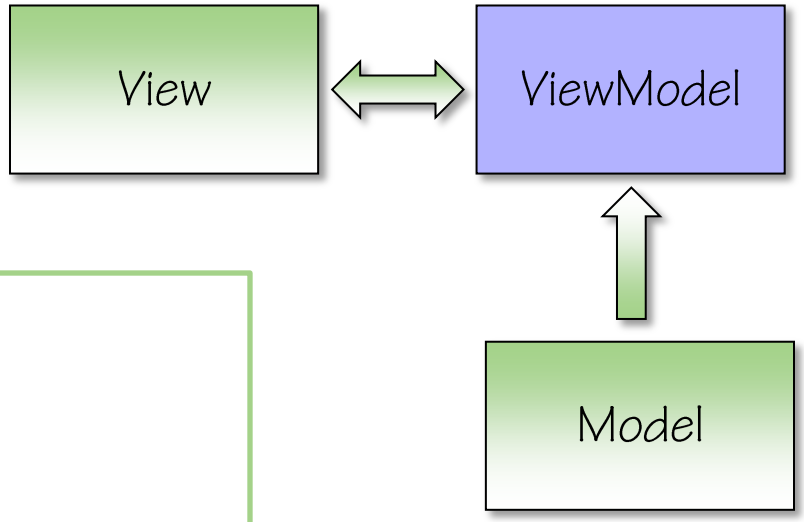
- Model
  - Has no references to the View or ViewModel
- View
  - May or may not have a reference to the ViewModel
  - Does not have a reference to the Model
- ViewModel
  - Has a reference to the Model
  - May or may not have a reference to the View

The Model

```
public class Person : INotifyPropertyChanged, IDataErrorInfo
{

    private string _firstName;
    public string FirstName
    {
        get { return _firstName; }
        set
        {
            _firstName = value;
            OnPropertyChanged("FirstName");
        }
    }

    private string _lastName;
    public string LastName
    {
        get { return _lastName; }
        set
        {
            _lastName = value;
            OnPropertyChanged("LastName");
        }
    }

...
```

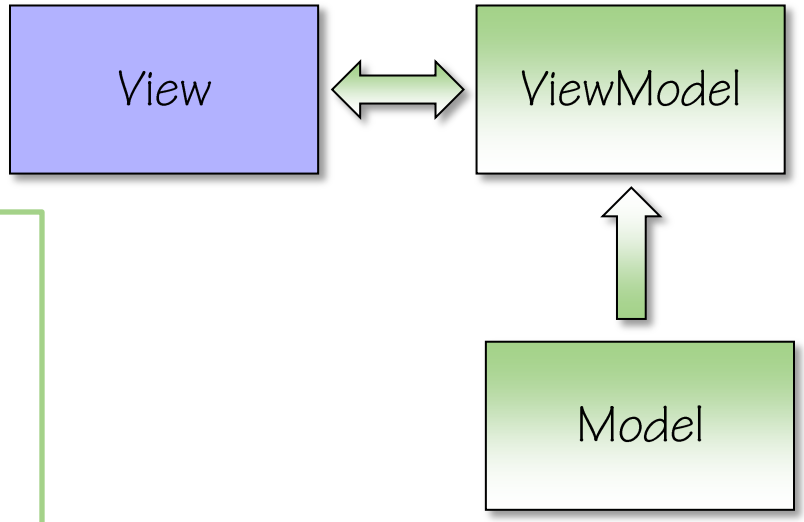


The ViewModel

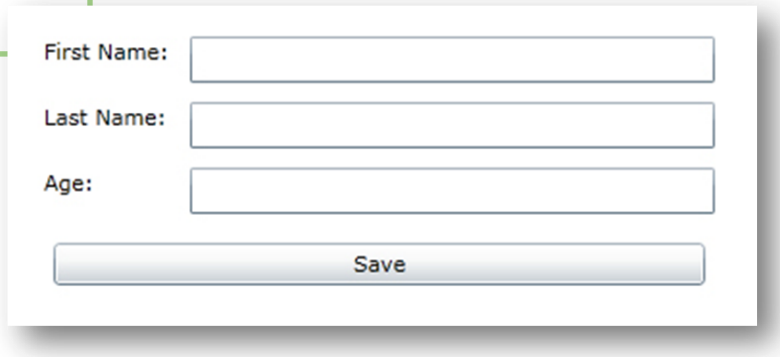



```
public class MainViewModel : INotifyPropertyChanged
{
    private Person _modelPerson;
    public Person ModelPerson
    {
        get { return _modelPerson; }
        set
        {
            _modelPerson = value;
            OnPropertyChanged("ModelPerson");
        }
    }
...
```

The View




```
<!-- First Name -->
<TextBlock Text="First Name:" Margin="5" />
<TextBox Grid.Column="1" Margin="5"/>

<!-- Last Name -->
<TextBlock Grid.Row="1" Text="Last Name:" Margin="5" />
<TextBox Grid.Row="1" Grid.Column="1"  Margin="5"/>

<!-- Age -->
<TextBlock Grid.Row="2" Text="Age:" Margin="5"/>
<TextBox Grid.Row="2" Grid.Column="1"  Margin="5"/>

<!-- Save Button -->
<Button Grid.Row="3" Grid.ColumnSpan="2" Content="Save" Margin="10"
```



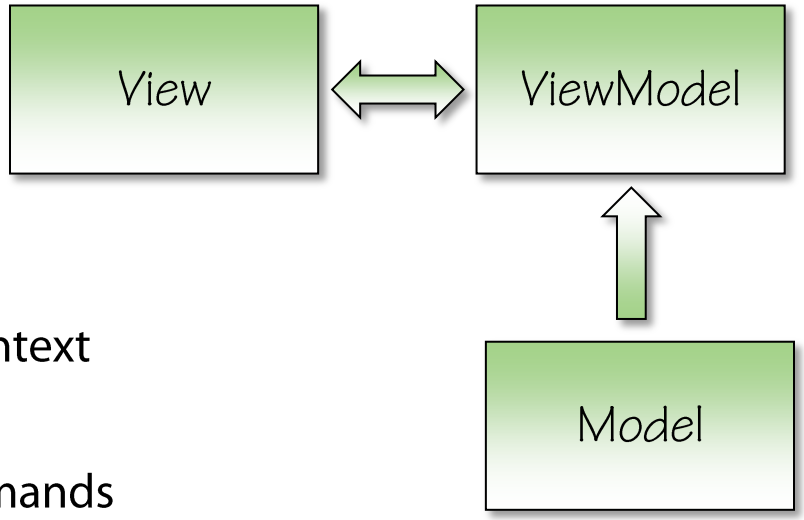
Binding the View to the ViewModel

- Declaratively in XAML
- Imperatively in Code
- ViewModel Locator
- Data Templates
- Inversion of Control (IoC)
- Factory Pattern with Inversion of Control
- and more…

Communication

- **Events**
- **Commands**
  - Execute
  - CanExecute

Collaboration



- **Model**
  - The data
- **View**
  - Binding to ViewModel set by the DataContext
- **ViewModel**
  - Exposes the Model as Properties or Commands
  - Must implement INotifyPropertyChanged

Consequences

- **Pro**
  - Reduce code-behind
  - Model doesn't need to change to support a view
  - Designers design, coders code
  - Reduces development time
  - Multi-targeting (project linking)
- **Con**
  - Create more files
  - Simple tasks can be complicated
  - Lack of standardization
  - Specific to WPF and Silverlight platforms

Known Uses

- **Microsoft**
- **UFC Gym**
- **US Army**
- **Family.Show**
- **PRISM Reference Implementation**
- **AQUA**
- **Many frameworks**
  - MVVM Light Toolkit
  - Caliburn
  - Cinch
  - Onyx
  - MVVM Foundation
  - And more…..

Related Patterns

- Model View Presenter (MVP)
- Model View Controller (MVC)
- Presentation Model (PM)

Summary

- Separate concerns
- Testability and Maintainability
- View and ViewModel binding
- Implementation