Hooking up Views and ViewModels in MVVM

Brian Noyes

CTO AND CO-FOUNDER, SOLLIANCE INC

@briannoyes www.briannoyes.com

Introduction

View-First Construction Patterns

Data Binding

ViewModel-First with DataTemplates

View-First XAML

```
In Code Behind: InitializeComponent()

<UserControl x:Class="..."
             xmlns="..."
             ...
             Width="300"
             Loaded="OnLoaded">
    <UserControl.DataContext>
        <cust:CustomerListViewModel />
    </UserControl.DataContext>
    <Grid>
        <DataGrid ItemsSource="{Binding Customers}" />
    </Grid>
</UserControl>
```

View-First Code-Behind

```
public partial class CustomerListView : UserControl
{
    public CustomerListView()
    {
        this.DataContext = new CustomerListViewModel();
        InitializeComponent();
    }
}
```

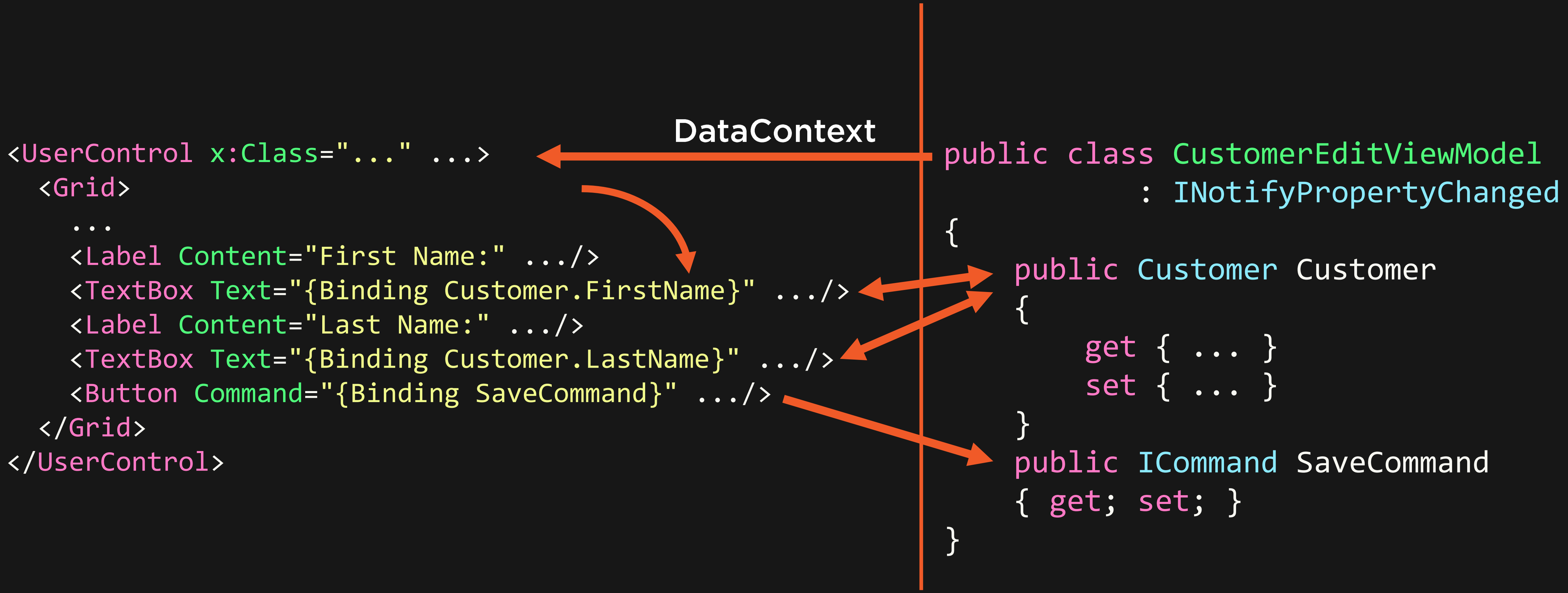
View-First: ViewModelLocator

ViewModelLocator is a meta-pattern for automatically locating and hooking up the right ViewModel

ViewModelLocator Process

- Determine which View type is being constructed
- Determine ViewModel type to create based on convention
- Construct ViewModel
- Set ViewModel as View's DataContext

Data Binding



```xml
<UserControl x:Class="..." ...>
  <Grid>
    ...
    <Label Content="First Name:" .../>
    <TextBox Text="{Binding Customer.FirstName}" .../>
    <Label Content="Last Name:" .../>
    <TextBox Text="{Binding Customer.LastName}" .../>
    <Button Command="{Binding SaveCommand}" .../>
  </Grid>
</UserControl>
```

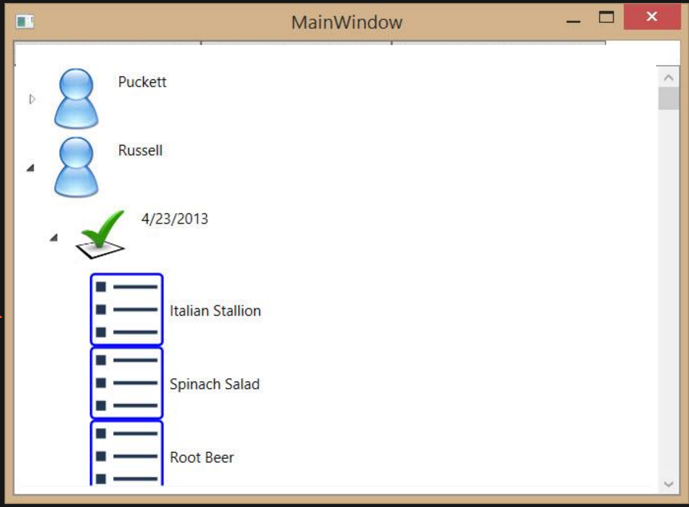
```csharp
public class CustomerEditViewModel
             : INotifyPropertyChanged
{
    public Customer Customer
    {
        get { ... }
        set { ... }
    }
    public ICommand SaveCommand
    { get; set; }
}
```

ViewModel-First with DataTemplates

CustomersOrderTreeViewModel

```
<Grid>
  <Grid.Resources>
    <DataTemplate DataType="{x:Type local:CustomersOrderTreeViewModel}">
      <local:CustomersOrderTreeView />
    </DataTemplate>
  </Grid.Resources>

  <ContentControl Content="{Binding CurrentViewModel}"
</Grid>
```

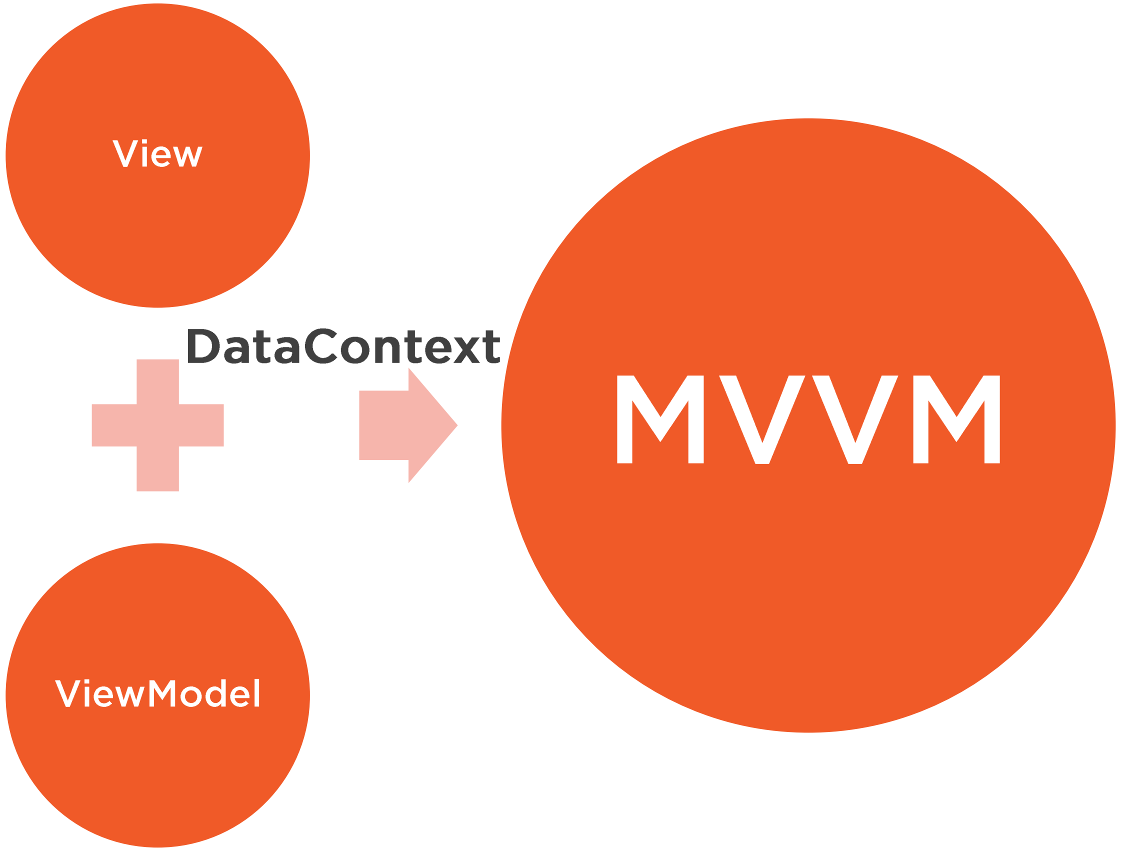


No One's on First

May have other code in control of constructing both View and ViewModel

Order does not matter

Set DataContext after constructing both



Summary

View-First MVVM hookup can be achieved several ways

Data binding forms the glue that flows

DataTemplates allow dynamic selection and hookup of ViewModels for Views