Commanding in Silverlight with mvvm

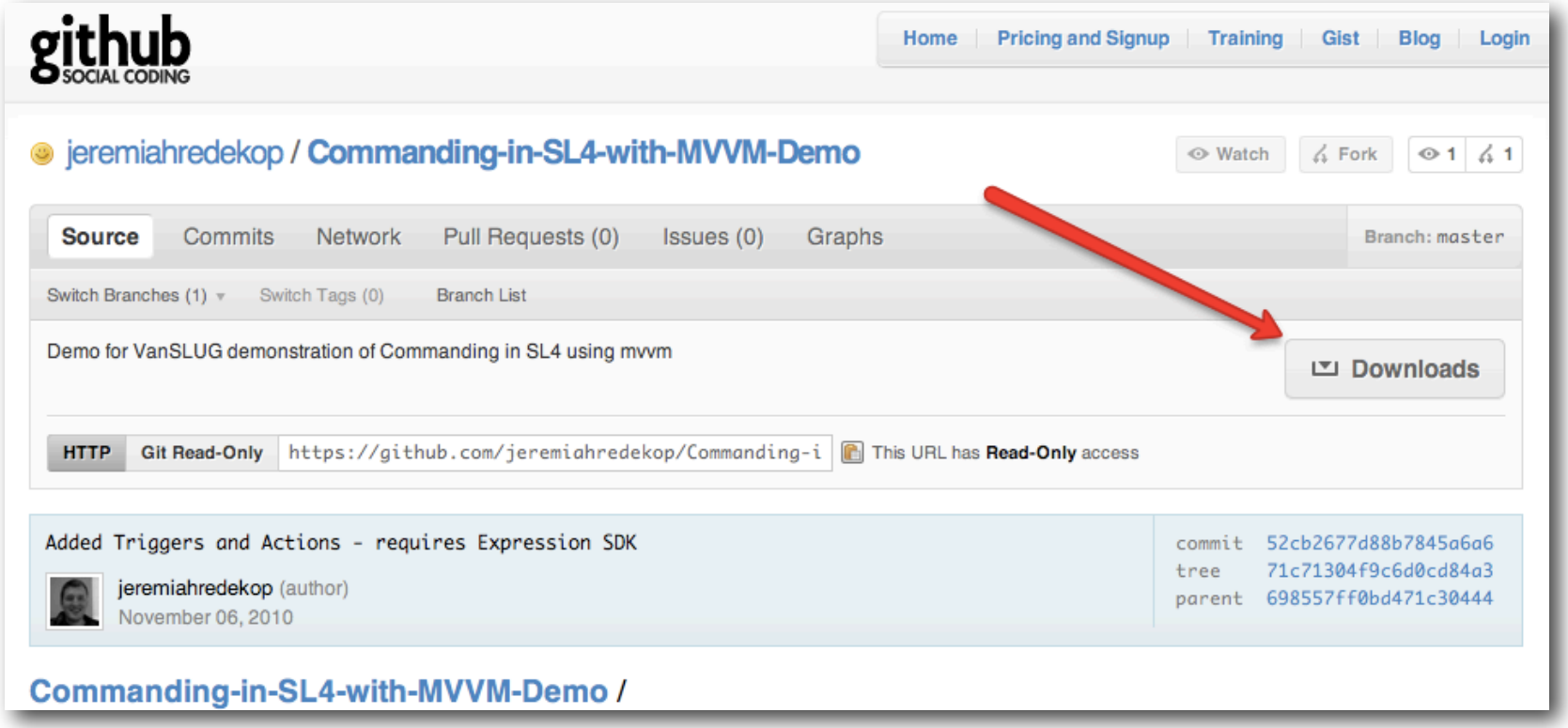
By Jeremiah Redekop
http://blogs.geniuscode.net/JeremiahRedekop
jeremiah@geniuscode.net

Evening Outline - 45min

- Mvvm Overview
  - ViewModel Locator
- Commanding Overview
- MVVMLight Framework
- Demo 1: Code-Behind
- Demo 2a Behaviors
- Demo 2b:Binding with Commanding
- Demo 3: Unit Testing ViewModel with Silverlight Test Framework
- Q&A

Get the Demo Code

- Available right now! Follow along!
- URL: http://bit.ly/bM2rZn

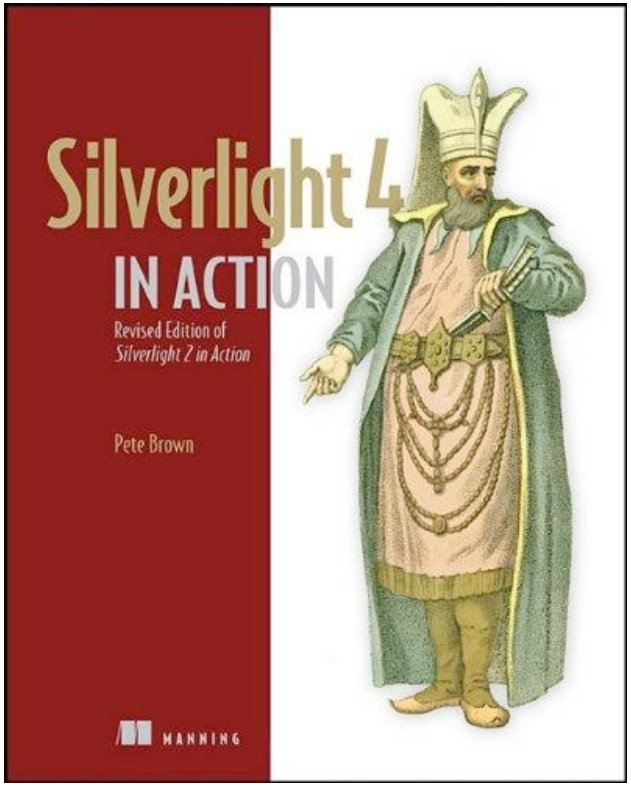


Silverlight 4 Resources

- Silverlight TV
- Book: Silverlight 4 In Action - Pete Brown
- Official Silverlight Forums
- Silverlight User Group (meetings & forums)

Post Questions now

- Twitter: #vanslug
- Forum: Event Announcement Forum
- eligible for Silverlight 4 In Action Book



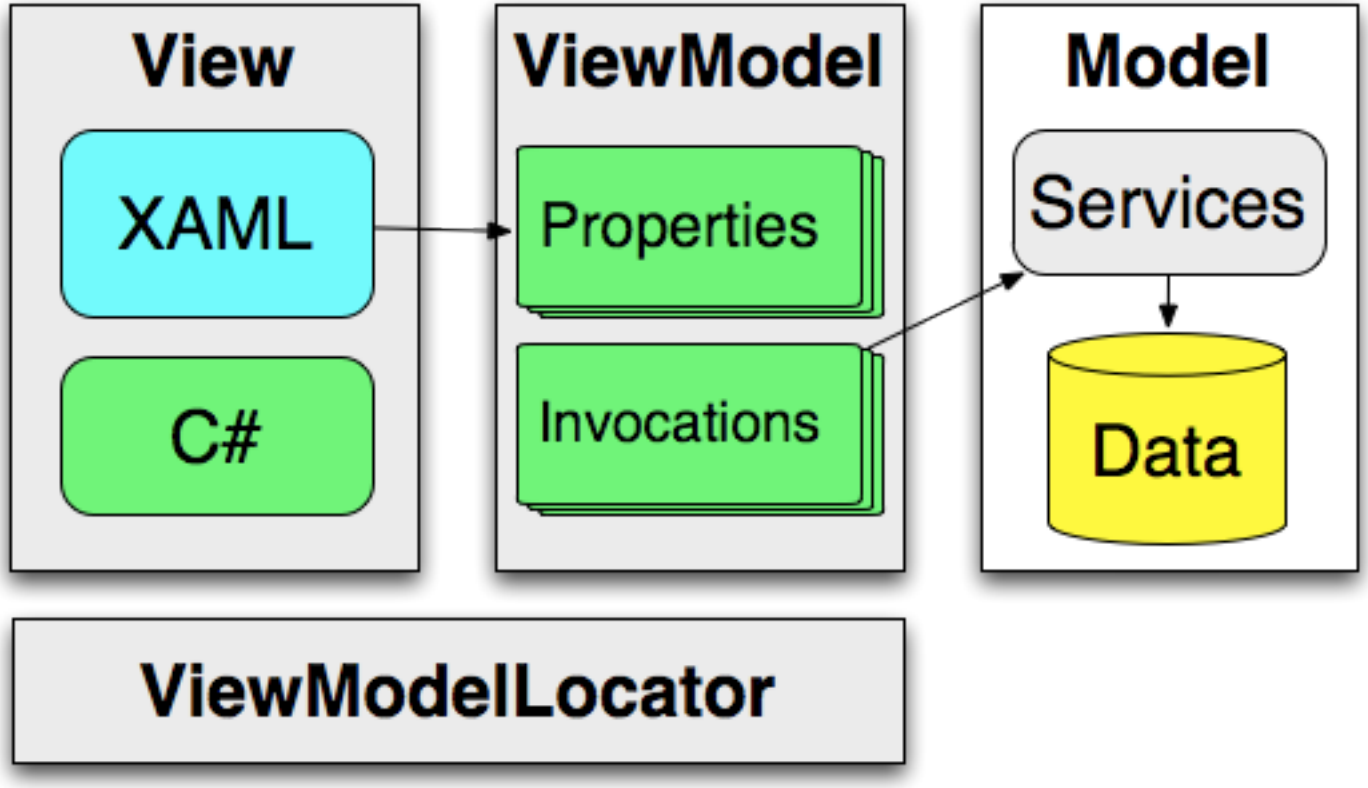
Important Disclaimer

- There is no best, or "right" way for MVVM!
- MVVM is just a pattern
- There are pros & cons to any approach: use what works
- Understand the underlying technologies

- Reality Check: your client doesn't care about mvvm

What is Mvvm?

- Separation of Concerns



What is the ViewModel?

- DataContext of the View
- Connection to Model
- Encapsulates & Isolates Logic

Where does ViewModel Come From?

- The View
  - Code
    - View's code behind creates ViewModel in constructor or event
  - Markup (XAML)
    - ViewModel is declared in the markup
    - Exactly the same result as code-behind
    - Allows for design time data preview
- External Source (ViewModel Locator)
  - Locator is declared in Markup, exposes ViewModel
  - Allows sharing of ViewModel between Views
  - ViewModel can outlive the View

ViewModel Locator as Resource

```
<Application x:Class="CommandingWithMvvm.App"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:vm="clr-namespace:CommandingWithMvvm.ViewModels">
    <Application.Resources>
        <ResourceDictionary>
            <vm:ViewModelLocator x:Key="VMLocator"/>
            <ResourceDictionary.MergedDictionaries>
                <ResourceDictionary Source="Assets/Styles.xaml"/>
            </ResourceDictionary.MergedDictionaries>
        </ResourceDictionary>
    </Application.Resources>
</Application>
```

Getting the ViewModel

- ViewModel set as DataContext in View, through the Resource:ViewModel Locator

```
<navigation:Page x:Class="CommandingWithMvvm.Binding1"
   DataContext="{Binding Binding1ViewModel, Source={StaticResource
VMLocator}}"
```

- ViewModel Locator exposes the ViewModel as a property

```
public Binding1ViewModel Binding1ViewModel
```

What is Commanding?

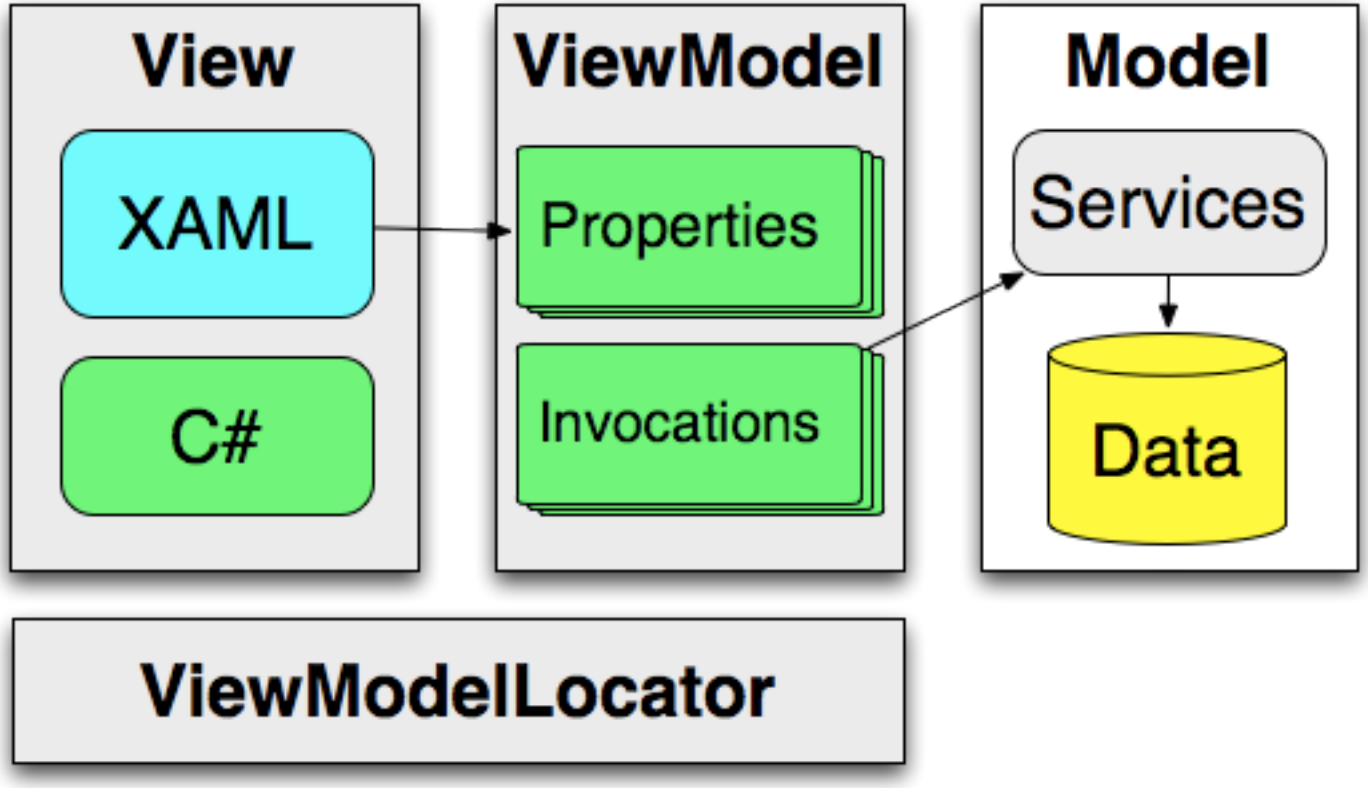
- Button bound to Property (ICommand) on DataContext
- No code-behind on view is required
- Mouse click invokes bound command
- **XAML**:

```
<Button Content="Bound to ViewModel Command"
Command="{Binding IncrementCount}"  />
```

- *IncrementCount is an proptery (ICommand) on the DataContext*

What is Commanding?

- ViewModel invokes the service via Command



The crux of Commanding

- ViewModel has logic to communicate with model
- ViewModel exposes commands that can be invoked
- View binds commands to UI Elements for the user to invoke

ICommand Interface

- Recently added in Silverlight 4
- Same contract as WPF

```
public interface ICommand
{
    event EventHandler CanExecuteChanged;
    bool CanExecute(object parameter);
    void Execute(object parameter);
}
```

RelayCommand

- MVVM Light - http://mvvmlight.codeplex.com
- Implementation of ICommand that relays functionality through lambdas

```
public class RelayCommand : ICommand
{
    public RelayCommand(Action execute, Func<bool> canExecute);
    public void RaiseCanExecuteChanged();
    public event EventHandler CanExecuteChanged;
    public bool CanExecute(object parameter);
    public void Execute(object parameter);
}
```

RelayCommand: Example

- Create Action to call method
- Create Func as predicate

- Create Command with delegates in constructor

```
private void InitializeCommands()
{
    // Action that command will perform
    Action incrementAction = () => PerformIncrementCount();
    // Predicate for whether or not action is allowed to execute
    Func<bool> incrementPredicate = () => Count >= 0;

    // Set Command property to new RelayCommand Object
    IncrementCount = new RelayCommand(incrementAction,incrementPredicate);
}
```

Demo

- Increment Number on the screen
- No Ria Services
- No Network Access
- Simple example to demonstrate:
  - commands
  - viewmodels

Demo 1: No ViewModel

- update values using c#
- View contains Variables
- View contains Logic
- Logic executed via event handlers

```
private void button1_Click(object sender, RoutedEventArgs e)
{
    //Update private variable
    count++;
    //Refresh Text Box from private value
    SetTextBoxValue();
}
```

Demo II: ViewModel

- ViewModel assigned through ViewModel Locator
- Textbox bound to property on ViewModel
- 2 parts:
  - Triggers
  - Commanding

Demo IIa: Triggers & Actions

- Mimicks code behind, but in XAML
- Requires Expression blend 4 SDK
  - Microsoft.Expression.Interactions.dll
  - System.Windows.Interactivity.dll

```
<Button Content="Behavior Button" >
    <i:Interaction.Triggers>
        <i:EventTrigger EventName="Click">
            <si:CallMethodAction MethodName="PerformIncrementCount"
            TargetObject="{Binding}" />
        </i:EventTrigger>
    </i:Interaction.Triggers>
</Button>
```

Demo IIb: MvvmLight Commanding

- Button bound to command on viewmodel

```
<Button Content="Bound to ViewModel Command"
Command="{Binding IncrementCount}"  />
```

Demo 3: Unit Testing

- Install Silverlight Toolkit April 2010
- Silverlight Unit Testing Framework
- All code executed on SL client

- Logic on ViewModel can be tested without UI

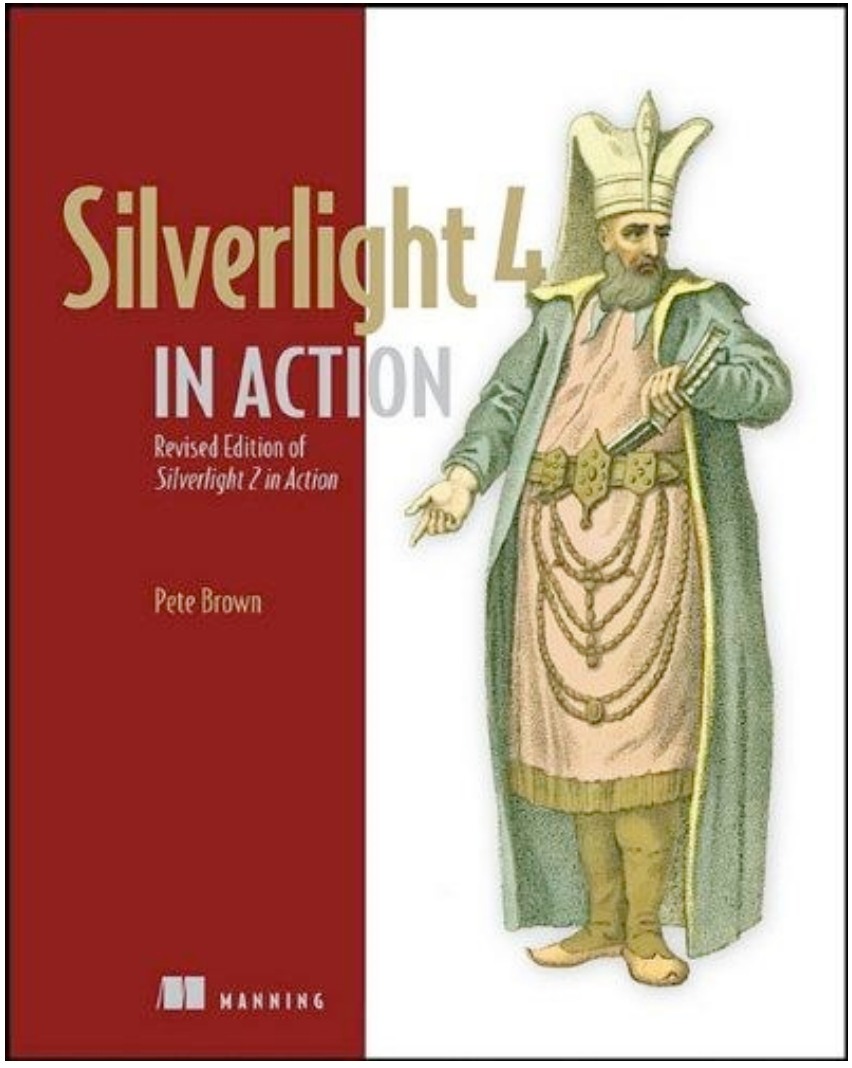
Summary

- Mvvm allows separation of logic & model activity from view
- Commanding & Behaviours allow for XAML invocation of ViewModel Code
- Commanding exposes bindable properties
- ViewModels can be unit tested

Q & A

- Twitter #vanslug
- Forums
- Live

- Silverlight 4 in action



- Thank you for your involvement!
- Please put your questions & feedback on the forum:
- http://forum.vanslug.net

Vanslug

- Thank you for your involvement!
- Please put your questions & feedback on the forum:
- http://forum.vanslug.net