Building Loosely Coupled XAML Client Apps with Prism

Brian Noyes
IDesign Inc. (www.idesign.net)
brian.noyes@idesign.net, @briannoyes

About Brian

Chief Architect
IDesign Inc. (www.idesign.net)

Microsoft Regional Director
(www.theregion.com)

Microsoft MVP
Silverlight

Publishing

Developers Guide to Microsoft Prism 4, O'Reilly & Assoc., March 2011

Developing Applications with Windows Workflow Foundation, LiveLessons training DVD, June 2007

Smart Client Deployment with ClickOnce, Addison Wesley, January 2007

Data Binding in Windows Forms 2.0, Addison Wesley, January 2006

The Silverlight Show, MSDN Magazine, MSDN Online, CoDe Magazine

Speaking

DevConnections, Microsoft TechEd, VSLive!, DevTeach, INETA User Groups

E-mail:brian.noyes@idesign.net
Twitter: @briannoyes
Blog:http://briannoyes.net

Prism

- Developed by Microsoft patterns and practices
- Old name:
  - Composite Application Guidance for WPF and Silverlight
- Guidance for building composite UI applications
  - Loosely coupled, modular, extensible applications
  - Based on well-known design patterns
- General design goals:
  - Light weight – use any combination of features
  - Use any container

Prism

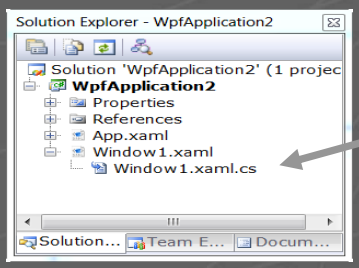
- Consists of:
  - Prism Library
    - The library formerly known as CAL
    - Really four assemblies
  - Stock Trader / MVVM Reference Implementation (RI)
  - Quickstarts
  - Documentation
  - How-Tos
- Current Released Version: 4.1
  - prism.codeplex.com

Prism Version History

- Prism 1 – June 2008 – WPF Only
- Prism 2 – Feb 2009 – Added support for Silverlight 3
- Prism 2.1 – October 2009 – Minor update
- Prism 2.2 – July 2010 – Updated for .NET 4 / Silverlight 4
- Prism 4 – November 2010 – MVVM / MEF / Navigation added
- Prism 4.1 – Feb 2012 – Silverlight 5 support
- Prism 4.5?

Why Composites?

- Good design – based on well known design patterns
- Loose coupling
- Separation of concerns
- End goal:
  - Maintainability
  - Extensibility
  - Testability
  - etc



Putting all your logic here is not good enough

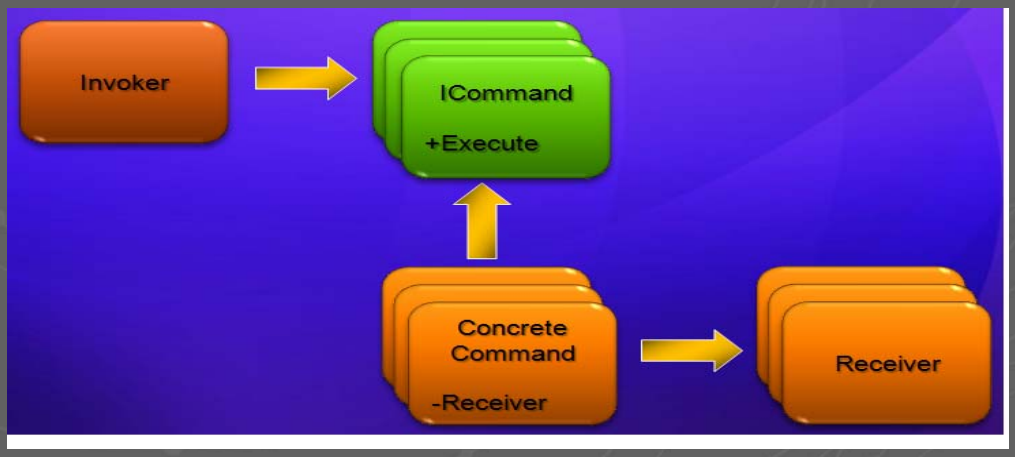
Prism Key Features

- App structure
  - Well defined abstractions for different kinds of presentation layer code
    - Model-View-ViewModel, Application Controller, Bootstrapper, Modules, etc
- Communications
  - Loosely coupled communications with events and commands
- Modularity
  - Compose application of loosely coupled units of functionality
- UI Composition
  - Compose the user interface of loosely coupled UI parts
- Navigation
  - Well defined approach for switching and navigating through different views/screens in the application

MVVM Guidance

- Documentation and samples on implementing the MVVM pattern
- Basic QuickStart, full QuickStart, and Reference Implementation sample
- Examples of:
  - Static hook up of view and view model from the view
  - Dynamic hook up of view and view model through data templates
  - ViewModel base class to encapsulate common concerns
  - “Wrapping” model properties to add property change and/or validation implementation
  - Exposing model properties when model supports needed interfaces
  - View model support for view that is not part of the model
  - Behaviors to communicate between view and view model

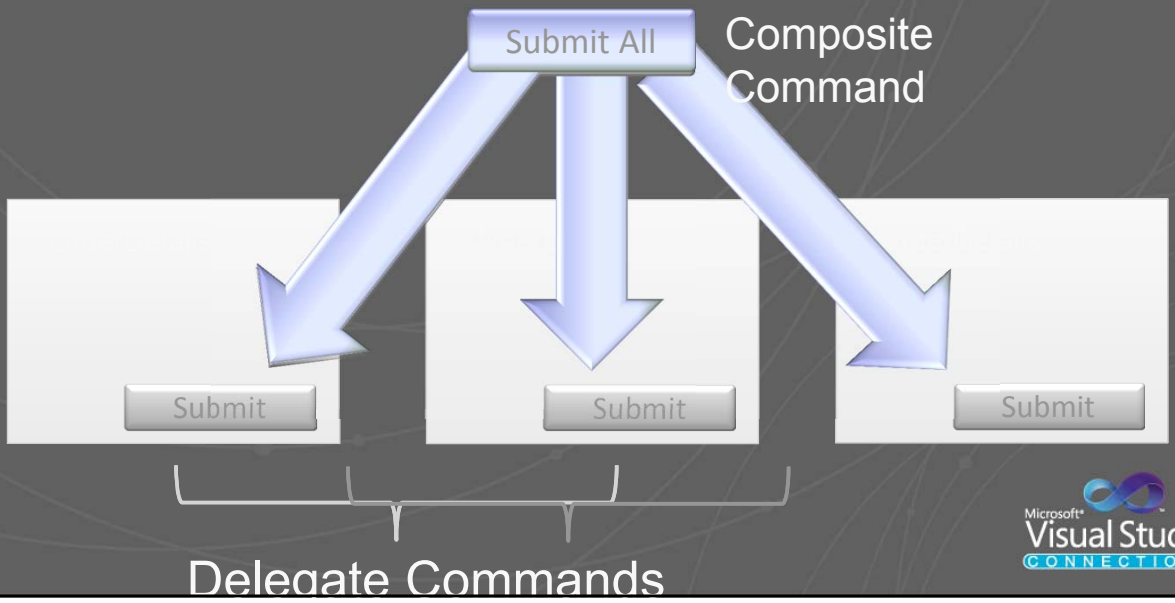
Command Pattern



Prism Commands

- Based on WPF/Silverlight ICommand interface
- Gets the handling out of the visual tree
  - Handlers can be view models or controllers
- Breaks the dependency on focus and event routing in the visual tree
- Allows multiple targets

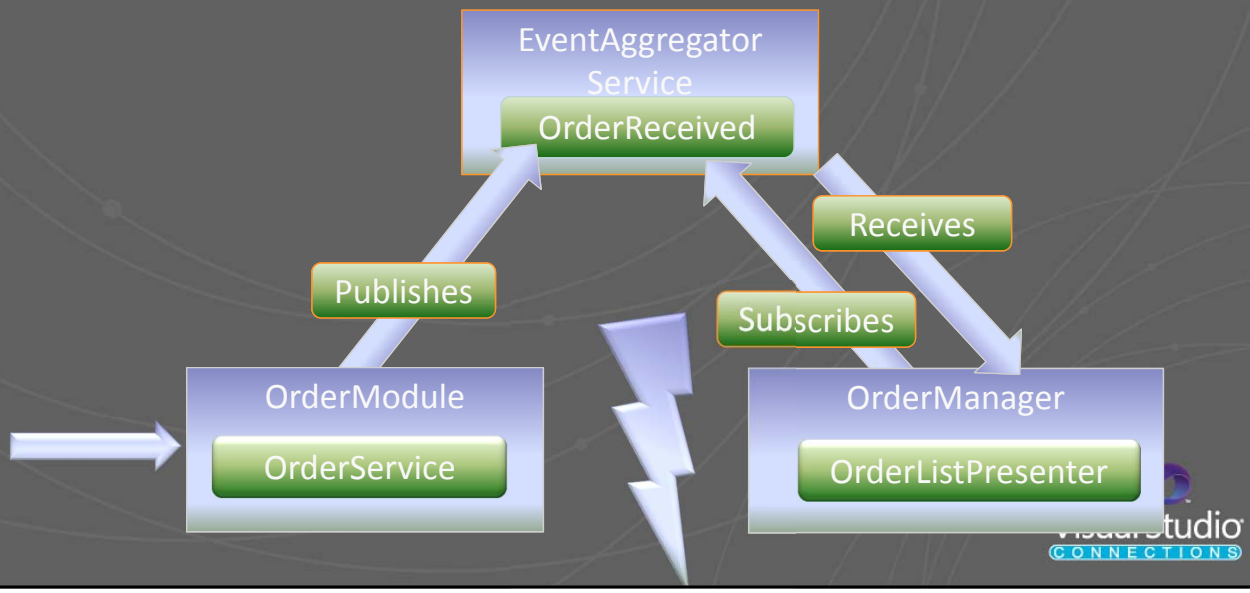
Composite Commands



Prism Events

- Address different scenarios than .NET Events or WPF/Silverlight Routed Events
- Based on pub/sub and event aggregator patterns
- Decouples publishers and subscribers
  - Type
  - Lifetime
- Prism provides IEventAggregator service
- Get event from aggregator
- Subscribe or publish
- Handles weak references
- Handles threading issues
- Provides filtering capability
- Allows communications between loosely coupled, non-visual parts
  - Presenters and controllers

Event Aggregation



Prism Application Architecture

- Prism Library
- Shell Application
  - Bootstrapper
  - Shell
  - Shared resources
- Modules
  - Views / ViewModels
  - Controllers
  - Services

Modules

- Unit of composition for the application
  - As opposed to views as the unit of composition for the UI
- Modules contain the artifacts for a portion of your application
  - Self-contained
  - Decoupled
  - Reusable
- Typically defined as a Visual Studio project / assembly
- Can have more than one module per assembly
  - Avoid

Modules

- Module class:
  - Initialize the objects in the module
- Like a Main() method for a library
- Known entry point in the module
- Initializes
  - Container types
  - Views
  - Regions
  - Services
  - Controllers
  - Etc

Module Loading

- Prism supports:
  - Statically from code
  - Dynamically
    - Module catalog XAML
    - Config file or directory scan in WPF
  - On-demand
- Modules can be dependent on other modules
  - Need to resolve load order in that case

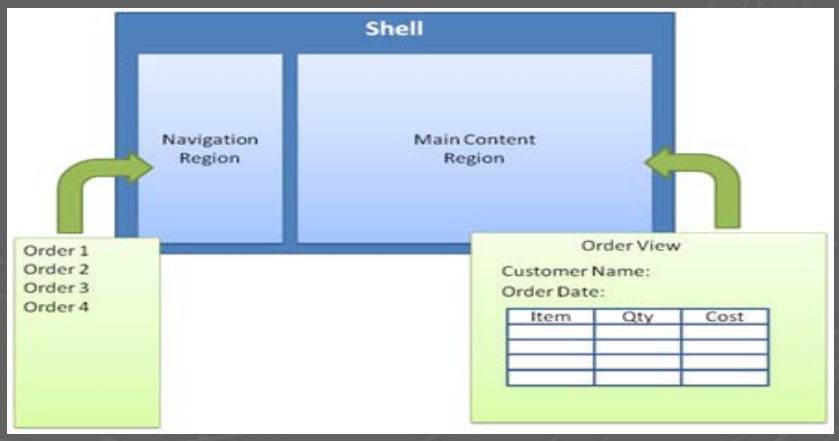
Modularity Implementations

- Two implementations in Prism 4:
  - Unity container
  - Managed Extensibility Framework (MEF)
- Either/or decision
- Same implementation patterns
- Difference is your use of dependency injection coding patterns in the module code

Regions

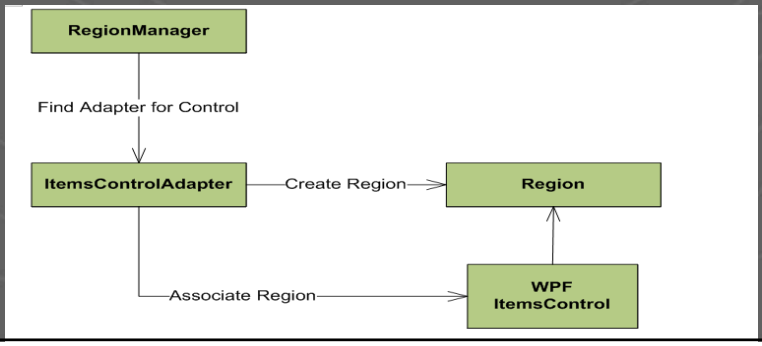
- Placeholder (named location) for view containment
  - Place to plug in views
  - Views dynamically added by app/module code
- Prism 2 supports two approaches:
  - View Discovery
  - View Injection
- Can be defined by the shell or a composite view
  - Shell – almost always
  - Composite View – less often

Regions



Region Manager

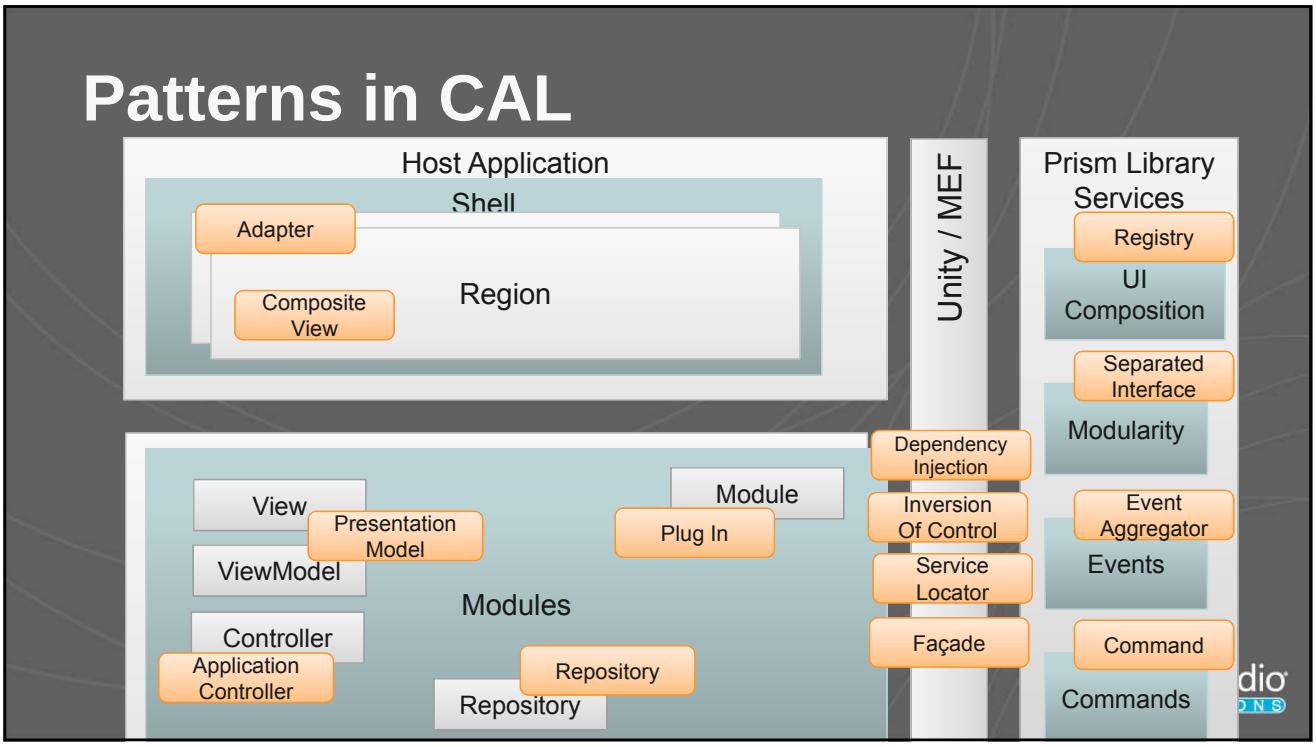
- Prism service
- Registration point for named regions
- Modules get a region references from the region manager
- Use the region to add their views



Navigation Guidance

- View-based Navigation
  - Use URIs to indicate logical views that you want to display/navigate to
  - Navigation service translates those URIs into view switching within a region
  - Journaling of where you have been with forward/back nav
  - Potential for deep linking in Silverlight
  - Integration with Silverlight Navigation framework
  - Allows views to decide whether to allow navigation in a standard way
    - i.e. unsaved work

Patterns in CAL



Resources

- Developers Guide to Microsoft Prism 4:
  - Electronic - http://msdn.microsoft.com/en-us/library/gg406140.aspx
  - Printed Book: http://shop.oreilly.com/product/0790145315496.do
- Working with Prism 4 series, Brian Noyes, The Silverlight Show,
  - http://www.silverlightshow.net/items/Working-with-Prism-4-Part-1-Getting-Started.aspx

E-mail:brian.noyes@idesign.net
Twitter: @briannoyes
Blog:http://briannoyes.net

Your Feedback is Important

Please fill out a session evaluation form drop it off at the conference registration desk.

Thank you!