Prism 4.1 Readme

Prism 4.1 - January 2011

WHAT'S NEW IN THIS RELEASE

Prism 4.1 is primarily focused on updating Prism for Silverlight 5 and addressing issues identified by the community on Codeplex.

Changes due to Silverlight 5 features:

- **UpdateTextBindingOnPropertyChanged** Blend behavior (Silverlight only) has been marked obsolete as Silverlight 5 now offers **UpdateSourceTrigger=PropertyChanged**.
- **DataTemplateSelector** (Silverlight only) has been marked obsolete as Silverlight 5 now offers a implicit data templates.
- **ButtonBaseClickCommand** has been marked obsolete.

Codeplex Issues:

- Codeplex Issue 5495 - Added **InvokeCommandAction**
- Codeplex Issue 5042 - Changed type constraint from **Control** to **UIElement** in WPF **CommandBehaviorBase** and **InvokeCommandActionClasses**.
- Codeplex Issue 8061: Added **SyncActiveStateAttribute** to sync activate state with scoped regions.
- Codeplex Issue 7234: **SubscriptionToken** is now disposable
- Codeplex Issue 3896: Added **ClearChildViewsRegionBehavior** and **ClearChildViews** attached property to optionally clear **RegionManager** on regions within a view.

Other Changes:

- Library and projects have been updated to reference the latest Unity and EnterpriseLibrary versions.
- A new NuGet package (4.1.0.0) is available that includes the binaries for .Net 4.0, Silverlight 4 and 5, and Windows Phone 7.1. Note that the Prism binaries for Silverlight 4 are the Prism 4.0.0.0 binaries.

Note: The documentation was not revised as part of this release so still refers to the 4.0 version, which remains accurate except for the changes listed above. Documentation on the new items above can be found later in this document.

WHAT'S INCLUDED IN THIS RELEASE

The following assets are shipped with Prism 4.1:

- Signed Prism Library for Windows Presentation Foundation (WPF) and Silverlight
- Signed Prism Library for Windows Phone 7
- Batch file to create custom Prism Library binaries
- Batch file to register the Prism Library binaries with Visual Studio
- Model-View-ViewModel Reference Implementation (MVVM RI)
- Stock Trader Reference Implementation (Stock Trader RI)
- Prism 4.0 documentation
- QuickStarts:
    - QuickStarts in Prism 4.0:
        - Basic MVVM QuickStart
        - MVVM QuickStart
        - Modularity QuickStarts (with MEF and Unity)
        - State-Based Navigation QuickStart
        - View-Switching Navigation QuickStart
        - UI Composition QuickStart
    - QuickStarts ported from Prism 2.x:
        - Commanding QuickStarts
        - Event Aggregation QuickStarts
        - Multi-Targeting QuickStarts
        - Hello World QuickStarts

With the exception of the MVVM, Navigation, and UI Composition QuickStarts, there are two solutions for each QuickStart, one that targets WPF and one that targets Silverlight.

This update does not update any of the documentation. The "What's New in Prism 4.0" in the Prism4.chm, pertains only to changes between Prism 2.1 to 4.0.

Note: For more information about using Prism on Windows Phone 7, see the Windows Phone 7 Developer Guide community site on CodePlex (http://wp7guide.codeplex.com/).

INSTALLING PRISM

This section describes how to install Prism. It involves the following three steps:

1. Install system requirements.
2. Extract the Prism source code, binaries, and documentation.
3. Register the Prism binaries.

Step 1: Install System Requirements

Prism was designed to run on the Microsoft Windows 7, Windows Vista, or Windows Server 2008 operating system. This version has been smoke tested on Windows XP Professional and Windows Server 2003, but it has not been exhaustively tested. WPF applications built using this guidance require the .NET Framework 4.0, and Silverlight applications require Silverlight 5.

Before you can use the Prism Library, the following must be installed:

- Microsoft .NET Framework 4.0 (installed with Visual Studio 2010)
- Microsoft Visual Studio 2010 Professional, Premium, or Ultimate editions

Note: Visual Studio 2010 Express Edition can be used to develop Prism applications using the Prism Library.

If you are developing Silverlight applications, the following must be installed:

- Microsoft Silverlight 5 Tools for Visual Studio 2010 (required for Silverlight development; includes the developer Silverlight runtime)

> **Note**: Although the Silverlight Tools for Visual Studio 2010 are not required, it is recommended for all WPF and Silverlight developers to download and use the latest version of the Silverlight Tools for Visual Studio 2010.
>
> The WPF and Silverlight Designer for Visual Studio is updated together with the Silverlight developer runtime and software development kit (SDK), which are included in the download. These updates come in the form of new features and bug fixes.

Optionally, you should consider also installing the following:

- Microsoft Expression Blend 4. A professional design tool for creating compelling user experiences and applications for WPF and Silverlight.
- Windows Phone SDK 7.1. For Windows Phone 7.1 development.

Step 2: Extract the Prism Source Code, Binaries, and Documentation

To install the Prism assets, right-click the Prismv41.exe file, and then click **Run as administrator**. This will extract the source code, binaries, and documentation into the folder of your choice.

Step 3: Register the Prism Library Binaries with Visual Studio

Registering the Prism Library with Visual Studio is not required, but for many developers, doing this simplifies referencing the Prism Library assets in their own projects. If you elect to register the binaries, they will be listed in the Visual Studio **Add References** dialog box when adding a reference. If you elect to not register the binaries, you will need to manually set a file reference to the Prism Library binaries in your projects. The Prism Library signed assemblies will be placed in the following folders:

- {prism}\Bin\Desktop
- {prism}\Bin\Silverlight
- {prism}\Bin\Phone

To register the Prism Library binaries, launch the RegisterPrismBinaries.bat batch file located in the folder where you extracted Prism. This batch file creates a temporary .reg file with the information required to register the Desktop, Silverlight, and Phone binaries and uses it to update the registry.

Because updating the registry is a privileged operation, a User Account Control (UAC) prompt will appear if you do not have elevated privileges. For additional information about UAC, see "What is User Account Control" at http://windows.microsoft.com/en-US/windows7/What-is-User-Account-Control.

Note: At most, one copy of the binaries can be registered using the script; if multiple copies of the Prism Library are registered only the binaries for the last registered copy will be available in Visual Studio.

DOCUMENTATION

Prism includes the following documentation:

- Prism4.chm is the guidance documentation.
- Prism4APIReference-Desktop.chm is the class library reference documentation for creating WPF applications with Prism.
- Prism4APIReference-Silverlight.chm is the class library reference documentation for creating Silverlight applications with Prism.
- Prism4APIReference-Phone.chm is the class library reference documentation for creating Windows Phone 7 applications with Prism.

Documentation Addendums for Prism 4.1

Using InvokeCommandAction

While Blend offers the **EventTrigger** and **InvokeCommandAction**, this action does not update the controls enabled state if the **ICommand** changes it. Prism's **InvokeCommandAction** will monitor the underlying ICommand's **CanExecuteChanged** event and update the control's enabled state appropriately. You can use it in place of Blend's **InvokeCommandAction**:

```
<TextBox>
  <i:Interaction.Triggers>
    <i:EventTrigger EventName="KeyUp">
        <prism:InvokeCommandAction Command="{Binding MyCommand}" />
    </i:EventTrigger>
  </i:Interaction.Triggers>
</TextBox>
```

Applying ClearChildViewsRegionBehavior

In some scenarios where a view contains nested region, the nested regions may need to be removed from the region manager when the parent view is removed from a region. To support this, an optional behavior was added that will clear the **RegionManager** attached properties on those regions to force their removal from the region manager.

While the behavior is registered and available for all regions, it is optionally enabled for a view containing nested regions by using the **ClearChildViews** attached property:

```
<UserControl x:Class="SomeModule.Views.RightView"
```

```
    ...
    xmlns:Regions="http://www.codeplex.com/prism"
    RegionBehaviors:ClearChildViewsRegionBehavior.ClearChildViews="True">
    ...
```

Syncing active to scoped regions

To support scenarios where a parent view contains scoped regions, the regions and views within that scope may need to be notified when the parent view becomes active. To handle this the SyncActiveStateAttribute can be applied to a View or ViewModel to ensure notification of the views (or view models) within the scoped regions.

```
[SyncActiveState]
public class MyViewModel : NotificationObject, IActiveAware
{
       (...)
}
```

QUICKSTARTS

The Prism4.chm contains details about each of the QuickStarts. The following information gives you an idea of the areas where Prism provides guidance and how to run the QuickStarts.

Basic MVVM QuickStart

The Basic MVVM QuickStart demonstrates how to build a very simple application that implements the MVVM pattern. This is provided to help you learn the basic concepts of the MVVM pattern.

The Basic MVVM QuickStart represents a subset of a survey application. A survey with different types of questions is shown; after the questionnaire is completed, it can be submitted. The user can also choose to reset the answers to the questions.

To run the Basic MVVM QuickStart:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Silverlight Only - Basic MVVM QuickStart.bat**
3. Set the BasicMVVMApp.Web project as the startup project.
4. Set BasicMVVMAppTestPage.html as the start page.
5. Press F5.

MVVM QuickStart

The MVVM QuickStart demonstrates how to build an application that implements the MVVM User Interface (UI) design pattern; it shows some of the more common programming tasks that developers can face, such as validation, UI interactions, and data templates.

The MVVM QuickStart represents a subset of a survey application. A survey with different types of questions is shown; after the questionnaire is completed, it can be submitted. The user can also choose to reset the answers to the questions.

To run the MVVM QuickStart:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Silverlight Only - MVVM QuickStart.bat**
3. Set the MVVM project as the startup project.
4. Press F5.

Modularity QuickStarts

The Modularity QuickStarts demonstrate how to code, discover, and initialize modules using Prism. These QuickStarts represent an application composed of several modules that are discovered and loaded in the different ways supported by the Prism Library using MEF and Unity as the composition containers. They also show how to use the new download progress features of Prism 4.0.

To run the WPF version of the Modularity QuickStart using MEF:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Desktop only - Open Modularity With Mef QuickStart.bat**
3. Set the ModularityWithMef.Desktop project as the startup project.
4. Press F5.

To run the WPF version of the Modularity QuickStart using Unity:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Desktop only - Open Modularity With Unity QuickStart.bat**
3. Set the ModularityWithUnity.Desktop project as the startup project.
4. Press F5.

To run the Silverlight version of the Modularity QuickStart using MEF:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Silverlight only - Open Modularity With Mef QuickStart.bat**
3. Set the ModularityWithMef.Silverlight.Web project as the startup project.
4. Press F5.

To run the Silverlight version of the Modularity QuickStart using Unity:

1. You must have completed the steps in Installing Prism.

2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Silverlight only - Open Modularity With Unity QuickStart.bat**
3. Set the ModularityWithUnity.Silverlight.Web project as the startup project.
4. Press F5.

State-Based Navigation QuickStart

The State-Based Navigation QuickStart demonstrates using Visual State Manager for application navigation. The QuickStart represents a subset of a chat application. After the application starts, a list of the user's contacts is displayed. Users can navigate to alternate views of their contacts: list, avatars, or contact details. As incoming messages arrive, they are displayed in a toast message. A contact's detail view provides the ability to send a message.

To run the State-Based Navigation QuickStart:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Silverlight only - Open QS - State-based Navigation QuickStart.bat**
3. Press F5.

View-Switching Navigation QuickStart

The View-Switching Navigation QuickStart demonstrates using region navigation APIs. The QuickStart is a simple email, contacts, and calendar application. The left region provides navigation to each of the views. The views demonstrate navigating backwards and asynchronous dialog interactions.

Note: This QuickStart does not demonstrate integrating region navigation with the native Silverlight navigation APIs. However, the Silverlight navigation APIs can be used with the Prism region navigation APIs.

To run the View-Switching Navigation QuickStart:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Silverlight only - Open QS - View-Switching Navigation QuickStart.bat**
3. Set the ViewSwitchingNavigation.Web project as the startup project.
4. Press F5.

UI Composition QuickStart

The UI Composition QuickStart illustrates how to use both the view discovery and view injection approaches for UI composition.

To run the UI Composition QuickStart:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Silverlight only - Open QS - UI Composition QuickStart.bat**

3. Right-click the **UIComposition.Web** project, and then click **Set as StartUp Project**.
4. Press F5.

Commanding QuickStart

The Commanding QuickStart demonstrates how to build a WPF or Silverlight UI that uses commands provided by the Prism Library to handle UI actions in a decoupled way.

To run the Commanding QuickStart:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click one of the following batch file to open the solution in Visual Studio: **Desktop & Silverlight - Open QS - Commanding QuickStart.bat** or **Desktop only - Open QS - Commanding QuickStart.bat**
3. Set the startup project. To build and run the WPF version of the QuickStart, the startup project should be the **Commanding.Desktop** project in the Desktop solution folder. To build and run the Silverlight version of the QuickStart, the startup project should be the **Commanding.Silverlight** project in the Silverlight solution folder.
4. Press F5.

Event Aggregation

The Event Aggregation QuickStart demonstrates how to build a composite application that uses the Event Aggregator service. This service enables you to establish loosely coupled communications between components in your application.

To run the Event Aggregation QuickStart:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click one of the following batch file to open the solution in Visual Studio: **Desktop & Silverlight - Open QS - EventAggregator QuickStart.bat** or **Desktop only - Open QS - EventAggregator QuickStart.bat**
3. Set the startup project. To build and run the Windows Presentation Foundation (WPF) version of the QuickStart, the startup project should be the **EventAggregation.Desktop** project in the Desktop solution folder. To build and run the Silverlight version of the QuickStart, the startup project should be the **EventAggregation.Silverlight** project in the Silverlight solution folder.
4. Press F5.

Multi-Targeting QuickStart

The Multi-Targeting QuickStart demonstrates the structure of a project created to multi-target WPF and Silverlight environments. It provides a desktop experience (on WPF) and a Rich Internet Application (RIA) experience (on Silverlight).

To run the Multi-Targeting QuickStart:

1. You must have completed the steps in Installing Prism.

2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Desktop & Silverlight - Open QS - MultiTargeting QuickStart.bat**
3. Set the startup project. To build and run the WPF version of the QuickStart, the startup project should be the **RealEstateListingViewer.Desktop** project in the Desktop solution folder. To build and run the Silverlight version of the QuickStart, the startup project should be the **RealEstateListingViewerHost** project in the Silverlight solution folder.
4. If you want to build and run the Silverlight version of the QuickStart, right-click the RealEstateListingViewerTestPage.html page, located in the **RealEstateListingViewerHost** project, and then click **Set As Start Page**.
5. Press F5.

Hello World QuickStarts

The Hello World QuickStarts are the ending solution for the hands-on labs "WPF Hands-On Lab: Get Started with the Prism Library" and "Silverlight Hands-On Lab: Get Started with the Prism Library" that are included in Appendix H in the Prism4.chm. In this lab, you will learn the basic concepts of Prism and apply them to create a Prism Library solution that you can use as the starting point for building a composite WPF or Silverlight application.

To run Hello World QuickStarts:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Desktop only - Open QS - Hello World QuickStart.bat** or **Silverlight only - Open QS - Hello World QuickStart.bat**.
3. Press F5.

REFERENCE IMPLEMENTATIONS

The Prism4.chm contains details about the reference implementations. The following information provides a summary and describes how to run the reference implementations.

MVVM RI

The MVVM RI application is a reference implementation that illustrates a complete survey application and demonstrates complex challenges that developers face when creating applications using the MVVM pattern. The main window shows a list of available questionnaires; when one is selected, an empty survey with different types of questions is shown. After the questionnaire is completed, it can be submitted. After that, the user is returned to the list of available questionnaires.

To run the MVVM RI:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click the following batch file to open the solution in Visual Studio: **Silverlight Only - MVVM Reference Implementation.bat**
3. Set the MVVM.Web project as the startup project.

4. Press F5.

Stock Trader RI

The Stock Trader RI application is a reference implementation that illustrates the baseline architecture. Within the application, you will see solutions for common, and recurrent, challenges that developers face when creating composite WPF applications.

The Stock Trader RI illustrates a fictitious, but realistic financial investments scenario. Contoso Financial Investments (CFI) is a fictional financial organization that is modeled after real financial organizations. CFI is building a new composite application to be used by their stock traders.

To run the Stock Trader RI:

1. You must have completed the steps in Installing Prism.
2. In Windows Explorer, double-click one of the following batch file to open the solution in Visual Studio: **Desktop & Silverlight - Open RI - StockTrader Reference Implementation.bat** or **Desktop only - Open RI - StockTrader Reference Implementaion.bat** (if you do not have Silverlight installed)
3. To run the WPF version of the Stock Trader RI, set the StockTraderRI project (located in the Desktop solution folder) as the startup project.
4. To run the Silverlight version of the Stock Trader RI, set the StockTraderRI project (located in the Silverlight solution folder) as the startup project.
5. Press F5.

MORE INFORMATION

Prism's community site is http://www.codeplex.com/Prism. On this community site, you can post questions, provide feedback, or connect with other users for sharing ideas.

UpdatePrismBinaries.bat

The purpose of the UpdatePrismBinaries.bat batch file is to help you automate rebuilding the Prism Libraries and copying them to the {prism}\Bin folder. If you are using the signed binaries that were included in the download, you will not need to use this batch file. However, if you are modifying the Prism source and want to rebuild and copy your modified assemblies to the {prism}\Bin folder, you can use this batch file to build and copy your assemblies.

Known Issues

- When running the Silverlight applications using the beta version of Internet Explorer 9, it has been found that Internet Explorer 9 will sometimes display an error message when an application first starts. If this occurs, click the Internet Explorer 9 **Refresh** button, and then the page will load correctly.
- After running the RegisterAssemblies.bat batch file, you may need to restart Visual Studio so that the Prism Library assembly references show up.

- In solutions that have both Silverlight and WPF projects, the Visual Studio Designer will display an error if the correct startup project is not selected. When viewing Silverlight code, set the Silverlight project as the startup project. When viewing WPF code, set the WPF project as the startup project.
- Running Code Analysis on the Prism Silverlight projects result in CA0055 warnings as FxCop cannot resolve references to older libraries. See connect issue 713608. The Prism library references versions of the Common Service Locator and Unity that still reference Silverlight 4.

Adding Prism Library Source Projects to Solutions

As part of shipping the Prism Library as signed binaries, the Prism Library projects were removed from the solutions of all QuickStarts and reference implementation projects. This was to make sure that the binaries were not accidentally overwritten when a solution was compiled. If you are a developer accustomed to stepping through the Prism Library code as you build your application, there are a couple of options:

- **Add the Prism Library Projects back in**. To do this, right-click the solution, point to **Add**, and then click **Existing project**. Select the Prism Library projects. Then, to prevent inadvertently building these, click **Configuration Manager** on the **Build** menu, and then clear the **Build** check box for all Prism Library projects in both the debug and release configurations.
- **Set a breakpoint and step in**. Set a break point in your application's bootstrapper, and then step in to a method within the base class (F11 is the typical C# keyboard shortcut for this). You may be asked to locate the Prism Library source code, but often, the full program database (PDB) file is available and the file will simply open. You may set breakpoints in any Prism Library project by opening the file and setting the breakpoint.

Prism Library and Code Access Security

The Prism Library uses all the default .NET Framework settings with respect to signing assemblies and code access security. It is a recommended practice to strong name all your assemblies, including the Prism Library assemblies, shell assembly, and any modules you might want to create. This is not a requirement. It is possible to load assemblies that have not been signed into a (signed or unsigned) Prism Library application. You can change this behavior by applying a .NET Framework security policy that does not allow the use of unsigned assemblies or one that changes the trust level of an assembly. Please note that the .NET Framework does not allow you to load partially trusted assemblies, unless you add the **AllowPartiallyTrustedCallers** attribute to the Prism Library assemblies.

For more information, see Code Access Security in the .NET Framework Developer's Guide on MSDN.