Advanced MVVM, Services and App Connect

In this chapter, we cover additional topics related to the programming model starting with more advanced coverage of the MVVM Architecture and data binding. The sample covers how to access RSS feeds, show a ProgressBar control from the ViewModel, how to lazy load images for better performance, and then concludes on how to data bind anything to anything using the IValueConverter Interface.

The following section covers advanced services starting with encryption services, an important topic and critical for most professionally developed applications. This is followed by a discussion covering advanced media services.

The rest of the chapter covers App Connect. App Connect refers to the various ways that applications can integrate with Windows Phone including integration with the pictures hub, the music+video hub, as well as integration with Bing Search. Next up we start with a discussion of Advanced MVVM programming topics.

Advanced MVVM

In this section, I cover how to access Syndicated Services like RSS feeds from Windows Phone, as well as advanced Model-View-ViewModel (MVVM) techniques to incorporate page navigation, showing progress, and lazy loading images. The last subsection covers the IValueConverter interface, which allows you to data bind any data type to just about any other data.

A sample project based on the MVVM Light project template named AdvancedMVVM is added to the Chapter 7 solution. MVVM is leveraged for this example (and in many samples throughout this chapter) to demonstrate how to handle slightly additional complexity of separating concerns between the View and the ViewModel when dealing with real-world scenarios.

By default, the MainPage.xaml binds to the MainViewModel in the ViewModel folder. MainPage.xaml presents a menu of four items that navigate to individual pages corresponding to the sections that follow:

- Syndicated Services (/View/SyndicatedServices.xaml)
- Showing Progress (/View/ShowProgress.xaml)
- Lazy Load Images (/View/LazyLoadImages.xaml)
- Data Bind to Anything (/View/DatabindToAnything.xaml)
MainPage.xaml consists of a ListBox data bound to a collection of items that represent the above pages. Here is the XAML:

```xml
<ListBox x:Name="listBox1" Margin="24,0,0,0"
    ItemsSource="{Binding Pages}"
    ItemTemplate="{StaticResource PageItemTemplate}" />
```

In MainViewModel.cs, the ApplicationTitle and PageName properties are added to the code. A PageItems property is added to MainViewModel class that is of type List<PageItem>:

```csharp
public const string PageItemsPropertyName = "PageItems";
private List<PageItemViewModel> _pages = null;
public List<PageItemViewModel> PageItems
{
    get
    {
        return _pages;
    }
    protected set
    {
        if (_pages == value)
        {
            return;
        }

        var oldValue = _pages;
        _pages = value;

        // Update bindings, no broadcast
        RaisePropertyChanged(PageItemsPropertyName);
    }
}
```

The MainViewModel class’s constructor takes an IPageItemsDataService Interface. MVVM Light then finds a concrete class registered via the inversion of control class SimpleIoc that implements that interface and provides the needed data service and data. I cover this in Chapter 7 in the MVVMLight sample. I recommend going through that section for more details on how MVVM Light works as well as the new Inversion of Control pattern.

Here is the constructor for MainViewModel:

```csharp
public MainViewModel(IPageItemsDataService dataService)
{
    _pageItemsService = dataService;
    PageItems = new List<PageItem>();

    _pageItemsService.GetPageItems(
        (pageItems, error) =>
        {
            if (error != null)
            {
                // Report error here
            }
        });
```