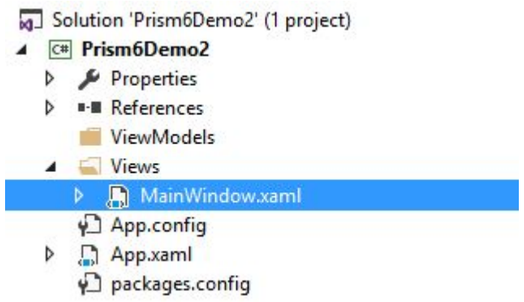
01-Creating shell

Repo: https://github.com/cmeegamarachchi/wpfprism-01-Shell

Create a Prism 6 based shell

1. Create a WPF application
2. Add **Prism.Wpf** and **Prism.Unity** packages via nu-get
3. Prism 6 viewmodel locator's default configuration requires views to be in a **Views** folder and view-models to be in **ViewModels** folder. So lets,
    a. Create two root level folders and name them **Views** and **ViewModels**
    b. Move MainWindow.xaml to Views folder
    c. Fix name spaces of MainWindow.xaml.cs and MainWindow.xaml to point to **Views** folder



4. As we are going to use a custom bootstrapper to start the application, lets remove startupUri from app.xaml

```
<Application x:Class="Prism6Demo2.App"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:local="clr-namespace:Prism6Demo2"
             StartupUri="Views/MainWindow.xaml">
    <Application.Resources>

    </Application.Resources>
</Application>

```

5. Create a custom bootstrapper
    a. Create a class at the root of the project and name it as Bootstrapper.cs
    b. Derive Bootstrapper class from UnityBootstrapper
    c. Add **Prism.Unity** and **Microsoft.Practices.Unity** namesspaces
    d. Override CreateShell and InitializeShell to tell the application to use MainWindow as the main shell

```
using System.Windows;
using Prism.Unity;
using Microsoft.Practices.Unity;
using Prism6Demo2.Views;

namespace Prism6Demo2
{
       public class Bootstrapper : UnityBootstrapper
       {
              protected override DependencyObject CreateShell()
              {
```

```
                        return Container.Resolve<MainWindow>();
                    }

                    protected override void InitializeShell()
                    {
                        Application.Current.MainWindow.Show();
                    }
                }
            }
```

6. Run the application