```
using GalaSoft.MvvmLight;
using System.Windows.Input;
using System.Collections;
using GalaSoft.MvvmLight.Command;
using Assignment6.Model;
using System.Drawing;
using System.Collections.ObjectModel;
using System.Net;
using System.Threading.Tasks;
using System;
using System.Xml.Linq;
using System.Linq;
using System.Collections.Generic;
using System.Windows;
using System.IO;

namespace Assignment6.ViewModel
{
    /// <summary>
    /// This class contains properties that the main View can data bind to.
    /// <para>
    /// Use the <strong>mvvminpc</strong> snippet to add bindable properties to this ViewModel.
    /// </para>
    /// <para>
    /// You can also use Blend to data bind with the tool's support.
    /// </para>
    /// <para>
    /// See http://www.galasoft.ch/mvvm
    /// </para>
    /// </summary>
    public class MainViewModel : ViewModelBase
    {
        public ICommand SearchCommand { get; private set; }
        public ICommand ShowImageCommand { get; private set; }
        private ObservableCollection<string> photoList;
        private ObservableCollection<string> titleList;
        private string selectedImage;
        private string searchText;
        private int selectedIndex;
        public WebClient flickrClient = new WebClient();
        Task<String> flickrTask = null;
        private const string KEY = "92df9b38a25b11ca3dab168760478b1c";
        /// <summary>
        /// Initializes a new instance of the MainViewModel class.
        /// </summary>
        public MainViewModel()
        {
            SearchCommand = new RelayCommand(SearchImagesMethodAsync);
            ShowImageCommand = new RelayCommand(DisplayImage);
            titleList = new ObservableCollection<string>();
            photoList = new ObservableCollection<string>();
        }

        public async void SearchImagesMethodAsync()
        {
            if (flickrTask != null && flickrTask.Status != TaskStatus.RanToCompletion)
            {
                var result = MessageBox.Show(
                    "Cancel the current Flickr search?",
                    "Are you sure?",
                    MessageBoxButton.YesNo,
                    MessageBoxImage.Question);
                if (result == MessageBoxResult.No)
                    return;
                else
                    flickrClient.CancelAsync();
            }
            else if (searchText == null || String.IsNullOrEmpty(searchText))
            {
                MessageBox.Show("You must enter search text!",
                    "Search Empty",
                    MessageBoxButton.OK,
                    MessageBoxImage.Error
                    );
            }
            else
            {
                var flickrURL = string.Format("https://api.flickr.com/services/rest/?method=flickr.photos.search&api_key={0}&tags={1}&tag_mode=all&per_page=500&privacy_filter=1&format=rest",
                    KEY,
                    searchText.Replace(" ", ","));
                titleList.Clear();
                photoList.Clear();
                selectedImage = null;
                RaisePropertyChanged("SelectedImage");
                RaisePropertyChanged("PhotoList");
                RaisePropertyChanged("TitleList");
                titleList.Add("Loading...");
                RaisePropertyChanged("TitleList");
                try
                {
                    flickrTask = flickrClient.DownloadStringTaskAsync(flickrURL);
                    XDocument flickrXML = XDocument.Parse(await flickrTask);
                    var flickrPhotos =
                        from photo in flickrXML.Descendants("photo")
                        let id = photo.Attribute("id").Value
                        let title = photo.Attribute("title").Value
                        let secret = photo.Attribute("secret").Value
                        let server = photo.Attribute("server").Value
                        let farm = photo.Attribute("farm").Value
                        select new FlickrResult
                        {
                            Title = title,
                            URL = string.Format(
                              "https://farm{0}.staticflickr.com/{1}/{2}_
                    {3}.jpg",
                                farm, server, id, secret)
                        };
                    titleList.Clear();
                    RaisePropertyChanged("TitleList");
                    if (flickrPhotos.Any())
                    {
                        var flickrPhotoList = flickrPhotos.ToList();
                        foreach (var p in flickrPhotoList)
                        {
                            photoList.Add(p.URL);
                        }
                        foreach (var t in flickrPhotoList)
                        {
                            titleList.Add(t.Title);
                        }
                        RaisePropertyChanged("TitleList");
                        RaisePropertyChanged("PhotoList");
                    }
                    else
                    {
                        titleList.Add("No matches");
                        RaisePropertyChanged("TitleList");
                    }
                }
                catch (WebException)
                {
                    if (flickrTask.Status == TaskStatus.Faulted)
                    {
                        MessageBox.Show("Unable to get results from Flickr",
                            "Flickr Error",
                            MessageBoxButton.OK,
                            MessageBoxImage.Error);
                    }
                }
            }

        }
```

```

        public void DisplayImage()
        {
            if (titleList.Count() == 1 && titleList.ElementAt(0).Equals("Loading..."))
            {
                //Do nothing
            }
            else if(selectedIndex >= 0 && selectedIndex < photoList.Count())
            {
                selectedImage = photoList.ElementAt(selectedIndex);
                RaisePropertyChanged( "SelectedImage");
            }
        }

        public ObservableCollection<string> TitleList
        {
            get { return titleList; }
            set
            {
                titleList = value;
                RaisePropertyChanged( "TitleList");
            }
        }

        public ObservableCollection<string> PhotoList
        {
            get { return photoList; }
            set
            {
                photoList = value;
                RaisePropertyChanged( "PhotoList");
            }
        }


        public string SearchText
        {
            get { return searchText; }
            set
            {
                searchText = value;
                RaisePropertyChanged( "SearchText");
            }
        }

        public string SelectedImage
        {
            get { return selectedImage; }
            set
            {
                selectedImage = value;
                RaisePropertyChanged("SelectedImage");
            }
        }

        public int SelectedIndex
        {
            get { return selectedIndex; }
            set
            {
                selectedIndex = value;
                RaisePropertyChanged("SelectedIndex");
            }
        }
    }
}
```

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace Assignment6
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
    }
}

```

```xml
<Window x:Class="Assignment6.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:i="http://schemas.microsoft.com/expression/2010/interactivity"
        xmlns:ei="http://schemas.microsoft.com/expression/2010/interactions"
        xmlns:local="clr-namespace:Assignment6"
        mc:Ignorable="d"
        Title="Flickr Viewer" Height="600" Width="750" ResizeMode="NoResize">
    <Grid DataContext="{Binding MainViewModel, Source={StaticResource ViewModelLocator}}" Background="LightGray">
        <ListBox ItemsSource="{Binding TitleList}" SelectedItem="{Binding SelectedImage}" SelectedIndex="{Binding SelectedIndex}" HorizontalAlignment="Left" Height="429" Margin="25,93,0,0" VerticalAlignment="Top" Width="180">
            <i:Interaction.Triggers>
                <i:EventTrigger EventName="MouseDoubleClick">
                    <i:InvokeCommandAction Command="{Binding ShowImageCommand}"/>
                </i:EventTrigger>
            </i:Interaction.Triggers>
        </ListBox>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="25,27,0,0" TextWrapping="Wrap" Text="{Binding SearchText}" VerticalAlignment="Top" Width="437"/>
        <Button Content="Search" Command="{Binding SearchCommand}" HorizontalAlignment="Left" Margin="495,27,0,0" VerticalAlignment="Top" Width="90" Height="23"/>
        <Border BorderThickness="1">
            <Image Source="{Binding SelectedImage}" HorizontalAlignment="Left" Height="429" Margin="220.2,92.2,0,0" VerticalAlignment="Top" Width="495"/>
        </Border>

    </Grid>
</Window>
```

```
/*
  In App.xaml:
  <Application.Resources>
      <vm:ViewModelLocator xmlns:vm="clr-namespace:Assignment6"
                           x:Key="Locator" />
  </Application.Resources>

  In the View:
  DataContext="{Binding Source={StaticResource Locator}, Path=ViewModelName}"

  You can also use Blend to do all this with the tool's support.
  See http://www.galasoft.ch/mvvm
*/

using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Ioc;
using Microsoft.Practices.ServiceLocation;

namespace Assignment6.ViewModel
{
    /// <summary>
    /// This class contains static references to all the view models in the
    /// application and provides an entry point for the bindings.
    /// </summary>
    public class ViewModelLocator
    {
        /// <summary>
        /// Initializes a new instance of the ViewModelLocator class.
        /// </summary>
        public ViewModelLocator()
        {
            ServiceLocator.SetLocatorProvider(() => SimpleIoc.Default);

            ////if (ViewModelBase.IsInDesignModeStatic)
            ////{
            ////    // Create design time view services and models
            ////    SimpleIoc.Default.Register<IDataService, DesignDataService>();
            ////}
            ////else
            ////{
            ////    // Create run time view services and models
            ////    SimpleIoc.Default.Register<IDataService, DataService>();
            ////}

            SimpleIoc.Default.Register<MainViewModel>();
        }

        public MainViewModel MainViewModel
        {
            get
            {
                return ServiceLocator.Current.GetInstance<MainViewModel>();
            }
        }

        public static void Cleanup()
        {
            // TODO Clear the ViewModels
        }
    }
}
```

```xml
<Application x:Class="Assignment6.App" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:local="clr-namespace:Assignment6" StartupUri="MainWindow.xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" d1p1:Ignorable="d" xmlns:d1p1="http://schemas.openxmlformats.org/markup-compatibility/2006">
  <Application.Resources>
    <ResourceDictionary>
      <vm:ViewModelLocator x:Key="ViewModelLocator" d:IsDataSource="True" xmlns:vm="clr-namespace:Assignment6.ViewModel" />
    </ResourceDictionary>
  </Application.Resources>
</Application>
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Assignment6.Model
{
    class FlickrResult
    {
        public string Title { get; set; }
        public string URL { get; set; }
    }
}

```