Lesson 7 Guide:
Working With Data

Table of Contents

Accessing Images in the User's Pictures Library

In the same manner, you can create an app that accesses the images stored in the users Pictures library. In the following project, you will create a viewer that displays thumbnails of the JPG and PNG documents in the user's Pictures library. The thumbnails will be shown in a single column GridView within a ScrollViewer. The user can click on a thumbnail to see a larger view of the image.

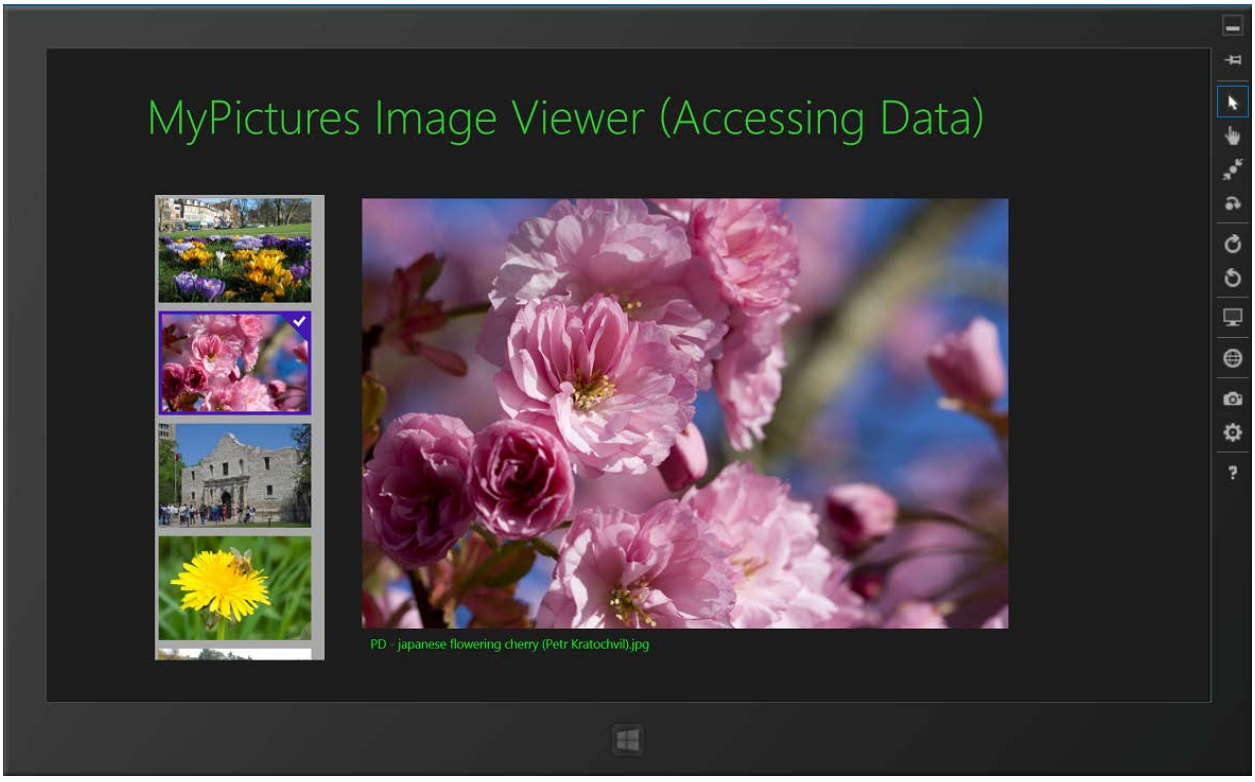

Figure 1 *– The Image Viewer accesses the user's Photo library.*

Start a new project using the Blank App (XAML) template. Modify the Mainpage.xaml document code as shown below.

XAML Code for Mainpage.xaml

```
<Page
    x:Class="_07_MyPictures_Data_Viewer.MainPage"
    DataContext="{Binding DefaultViewModel, RelativeSource={RelativeSource Self}}"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:_07_MyPictures_Data_Viewer"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">

    <Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}">
        <Grid Style="{StaticResource LayoutRootStyle}" >
            <Grid.RowDefinitions>
                <RowDefinition Height="140"/>
```

```
                <RowDefinition Height="*"/>
            </Grid.RowDefinitions>
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="120"/>
                    <ColumnDefinition Width="*"/>
                </Grid.ColumnDefinitions>
                <TextBlock x:Name="pageTitle" Grid.Column="1"
                      Text="MyPictures Image Viewer (Accessing Data)"
                      Style="{StaticResource PageHeaderTextStyle}"
                      Foreground="LimeGreen"/>
            </Grid>
            <ScrollViewer HorizontalAlignment="Left" Height="549" Margin="128,33,0,0"
                 Grid.Row="1" VerticalAlignment="Top" Width="200"
                 VerticalScrollMode="Enabled" HorizontalScrollMode="Disabled">
                <GridView x:Name="lvThumbs" Background="DarkGray"
                    SelectionChanged="ThumbnailSelectionChanged" >
                    <GridView.ItemTemplate>
                        <DataTemplate>
                            <Grid>
                                <Image Width="180" Source="{Binding}"
                                     Stretch="UniformToFill" ></Image>
                            </Grid>
                        </DataTemplate>
                    </GridView.ItemTemplate>
                </GridView>
            </ScrollViewer>
            <TextBox x:Name="txtInfo" HorizontalAlignment="Left" Height="24"
                  Margin="372,550,0,0" Grid.Row="1" TextWrapping="Wrap" Text="TextBox"
                  VerticalAlignment="Top" Width="954" BorderThickness="0"
                  Background="{x:Null}" Foreground="#FF0AF71C"/>
            <Image x:Name ="imgTest" HorizontalAlignment="Left" Height="508"
                  Margin="372,37,0,0" Grid.Row="1" VerticalAlignment="Top" Width="954"/>
        </Grid>
    </Grid>
</Page>
```

In the code behind (Mainpage.xaml.cs or Mainpage.xaml.vb), the StorageFolder object is set to KnownFolders.PicturesLibrary.

C# Code (Mainpage.xaml.cs)

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using Windows.Foundation;
using Windows.Foundation.Collections;
using Windows.UI.Xaml;
using Windows.UI.Xaml.Controls;
using Windows.UI.Xaml.Controls.Primitives;
using Windows.UI.Xaml.Data;
using Windows.UI.Xaml.Input;
using Windows.UI.Xaml.Media;
using Windows.UI.Xaml.Navigation;

using Windows.Storage;
using System.Collections.ObjectModel;
```

```
using Windows.Storage.FileProperties;
using Windows.UI.Xaml.Media.Imaging;
using Windows.Storage.Streams;

namespace _07_MyPictures_Data_Viewer
{

    public sealed partial class MainPage : Page
    {
        StorageFolder myPicsFolder = KnownFolders.PicturesLibrary;
        //Add:  using System.Collections.ObjectModel;
        ObservableCollection<BitmapImage> myImages = new
                                ObservableCollection<BitmapImage>();
        List<String> filenames = new List<String>();

        public MainPage()
        {
            this.InitializeComponent();
        }


        protected async override void OnNavigatedTo(NavigationEventArgs e)
        {
            foreach (StorageFile xyz in await myPicsFolder.GetFilesAsync())
            {
                //Add:  using Windows.UI.Xaml.Media.Imaging;
                BitmapImage img = new BitmapImage();
                //Add:  using Windows.Storage.FileProperties;
                img.SetSource(await xyz.GetThumbnailAsync(ThumbnailMode.PicturesView));
                if (xyz.Path.EndsWith("jpg") || xyz.Path.EndsWith(".png"))
                {  // only show JPG and PNG files
                    myImages.Add(img);
                    filenames.Add(xyz.Name);
                }
                if (myImages.Count > 0)
                {
                    DisplayImage(0);  //show the first image in list
                }
            }
            lvThumbs.ItemsSource = myImages;
            txtInfo.Text = lvThumbs.Items.Count.ToString();

        }

        private void ThumbnailSelectionChanged(object sender, SelectionChangedEventArgs
e)
        {
            int index = lvThumbs.SelectedIndex; //get which thumbnanil was selected
            DisplayImage(index);
        }

        private async void DisplayImage(int idx)
        {
            txtInfo.Text = filenames[idx];  //Show the file name below the image
            //Get the file using the reference from the filenames List
            StorageFile file = await
                          KnownFolders.PicturesLibrary.GetFileAsync(filenames[idx]);
            BitmapImage bitmap = new BitmapImage();  //create a new bitmap
```

```
            //Open a filestream to the bitmap
            // Add:  using Windows.Storage.Streams;
            IRandomAccessStream fileStream = await
                        file.OpenAsync(Windows.Storage.FileAccessMode.Read);
            bitmap.SetSource(fileStream);
            //set the source for the Image XAML control
            imgTest.Source = bitmap;
        }
    }
}
```

VB Code (Mainpage.xaml.vb)

```
Imports Windows.Storage
Imports System.Collections.ObjectModel
Imports Windows.Storage.FileProperties
Imports Windows.UI.Xaml.Media.Imaging
Imports Windows.Storage.Streams

Public NotInheritable Class MainPage
    Inherits Page

    Dim myPicsFolder As StorageFolder = KnownFolders.PicturesLibrary
    'Add:  imports System.Collections.ObjectModel
    Dim myImages As ObservableCollection(Of BitmapImage) = New _
                            ObservableCollection(Of BitmapImage)()
    Dim filenames As List(Of String) = New List(Of String)()


    Protected Overrides Async Sub OnNavigatedTo(e As Navigation.NavigationEventArgs)

        Dim xyz As StorageFile
        For Each xyz In Await myPicsFolder.GetFilesAsync()
            'Add:  imports Windows.UI.Xaml.Media.Imaging
            Dim img As BitmapImage = New BitmapImage()
            'Add:  using Windows.Storage.FileProperties;
            img.SetSource(Await xyz.GetThumbnailAsync(ThumbnailMode.PicturesView))
            If (xyz.Path.EndsWith("jpg") Or xyz.Path.EndsWith(".png")) Then
                ' only show JPG and PNG files
                myImages.Add(img)
                filenames.Add(xyz.Name)
            End If
            If (myImages.Count > 0) Then
                DisplayImage(0)   'show the first image in list
            End If
        Next
        lvThumbs.ItemsSource = myImages
        txtInfo.Text = lvThumbs.Items.Count.ToString()
    End Sub

    Private Async Sub DisplayImage(ByVal idx As Integer)
        txtInfo.Text = filenames(idx)   'Show the file name below the image
        'Get the file using the reference from the filenames List
        Dim file As StorageFile = Await _
                    KnownFolders.PicturesLibrary.GetFileAsync(filenames(idx))
        Dim bitmap As BitmapImage = New BitmapImage()  'reate a new bitmap
        'Open a filestream to the bitmap
        'Add:  imports Windows.Storage.Streams
```

```
        Dim fileStream As IRandomAccessStream = Await _
                        file.OpenAsync(Windows.Storage.FileAccessMode.Read)
        bitmap.SetSource(fileStream)
        ' set the source for the Image XAML control
        imgTest.Source = bitmap
    End Sub

    Private Sub ThumbnailSelectionChanged(sender As Object, _
                        e As SelectionChangedEventArgs) Handles lvThumbs.SelectionChanged
        Dim index As Integer = lvThumbs.SelectedIndex  'get which thumbnail was selected
        DisplayImage(index)
    End Sub
End Class
```

Remember to add the Pictures Library to the Capabilities list in Package.appxmanifest.

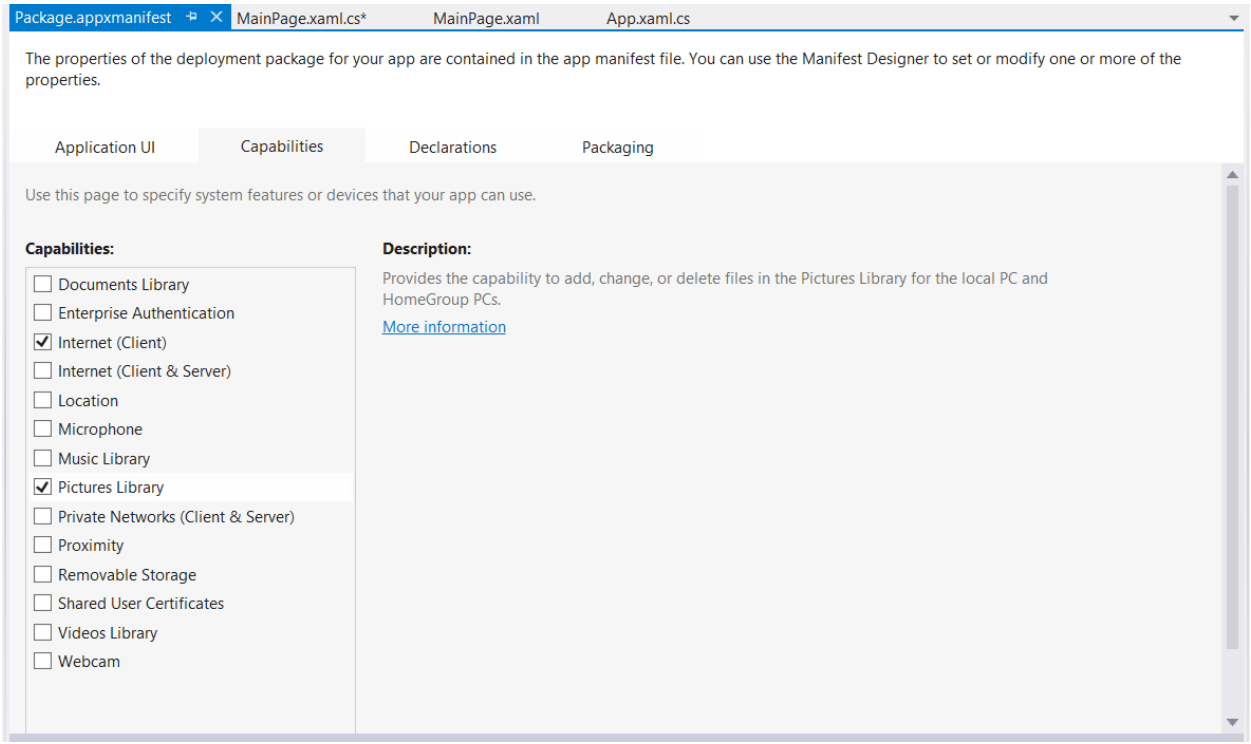

Figure 2 *– Adding access to the user's Pictures Library in the Package.appxmanifest file.*

Using JSON Formatted Data

JSON (JavaScript Object Notation), like XML (Extensible Markup Language), is a text-based open standard for data interchange. It is human-readable and language independent, utilizing key/value pairs. Like XML, its origin was in transmitting data between a web application and a server, but JSON also is useful as a back-end data source for both local and cloud apps. JSON data is transmitted as a string, which is serialized and de-serialized by a program. JSON is quickly gaining popularity due to its simplicity of use and readability in comparison to XML. For Windows 8, a JSON.NET package can be added to a project. The JSON.NET package makes serialization (translating an object-oriented data structure to a storage format) and de-serialization (translating a data storage format into an object-oriented data structure) straight forward and simple from a coding perspective.

The format of a JSON data file is of key:value pairs. The following is a sample JSON file showing two records for U.S. president information. Keys are lastName, firstName, number, yearsInOffice, inaugurated, inaugurationAge, homeState, rating, politicalParty, occupation, and college.

Note the following JSON formatting elements:

- Each record is enclosed in curly brackets.
- Records are comma-delimited.
- All the data (entire file) is enclosed by square brackets.
- Key names are enclosed in quotes.
- A colon separates the key and its value.
- Each key:value pair is comma delimited.


```
[{"lastName":"Roosevelt",
"firstName":"Theodore 'Teddy'",
"number":26,
"yearsInOffice":8,
"inaugurated":1901,
"inaugurationAge":42,
"homeState":"New York",
"rating":810,
"politicalParty":"Republican",
"occupation":"Author",
"college":"Harvard"
},
{"lastName":"Taft",
"firstName":"William Howard",
"number":27,
"yearsInOffice":4,
"inaugurated":1909,
"inaugurationAge":51,
"homeState":"Ohio",
"rating":491,
"politicalParty":"Republican",
"occupation":"Lawyer",
"college":"Yale"
}]
```


The data used in the following JSON example follows this format, but includes records for all the U.S. presidents of the twentieth century.

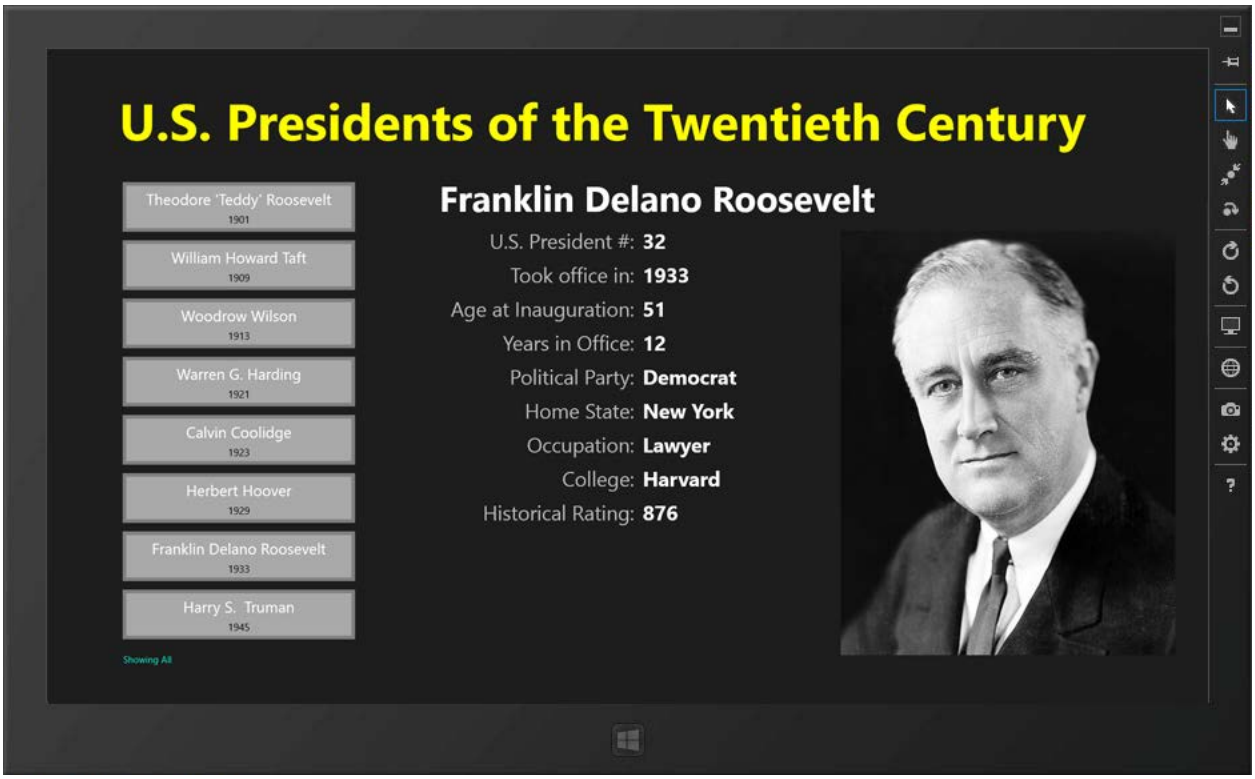

Figure 3 *– The information for each president is stored in a JSON-formatted text data file.*

This project was created from the Blank template. The XAML for the MainPage.xaml document consists of the following:

- A ListView on the left
- A series of TextBlocks to the right as labels and data-bound text displays
- An Image control on the far right

There is also a very small TextBlock at the bottom-left corner that shows whether all presidents are displayed in the ListView, just the Democrats, or just the Republicans. These are filtered via a bottom app bar. A DataTemplate is defined for how the record links are formatted in the ListView.

In the following code listing, named controls are highlighted in yellow, data bindings in cyan, and events in green.

XAML code (MainPage.xaml)

```
<Page
    x:Class="_07_JSON_Demo_1.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:_07_JSON_Demo_1"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">

    <Page.Resources>
        <DataTemplate x:Key="PrezOverview">
            <Border BorderBrush="Gray" BorderThickness="4">
                <Grid Height="50" Width="300" Background="DarkGray"
```

```
                        HorizontalAlignment="Center">
                    <StackPanel VerticalAlignment="Center" Width="300"
                                HorizontalAlignment="Left">
                      <TextBlock Height="28" FontSize="18" TextAlignment="Center"
                                Width ="300" Foreground="White" Text="{Binding
                                fullName}" />
                      <TextBlock Height="15" FontSize="12" TextAlignment="Center"
                                Foreground="Black" Text="{Binding inaugurated}" />
                    </StackPanel>
                </Grid>
            </Border>
        </DataTemplate>
    </Page.Resources>


    <Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}">
        <TextBlock HorizontalAlignment="Left" Height="86" Margin="85,50,0,0"
                TextWrapping="Wrap" Text="U.S. Presidents of the Twentieth Century"
                VerticalAlignment="Top" Width="1214" Foreground="Yellow" FontSize="60"
                FontWeight="Bold"/>
        <Grid HorizontalAlignment="Left" Height="520" Margin="85,177,0,0"
                VerticalAlignment="Top" Width="300">
            <ListView x:Name="lvPresidents" ItemTemplate="{StaticResource PrezOverview}"
                  ItemClick="PresidentChosen" IsItemClickEnabled="True"
                  Margin="0,-28,0,0"/>
        </Grid>

        <Grid x:Name="gridTest" HorizontalAlignment="Left" Height="600"
                Margin="453,141,0,0" VerticalAlignment="Top" Width="900">
            <TextBlock x:Name="txtPrezFullName" HorizontalAlignment="Left" Height="56"
                  Margin="10,10,0,0" TextWrapping="Wrap" Text="{Binding fullName}"
                  VerticalAlignment="Top" Width="830" FontSize="42" FontWeight="Bold"/>
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,70,0,0"
                  TextWrapping="Wrap" Text="U.S. President #:" VerticalAlignment="Top"
                  Width="240" TextAlignment="Right" FontSize="24"
                  Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezNumber" HorizontalAlignment="Left" Height="32"
                  Margin="250,70,0,0" TextWrapping="Wrap" Text="{Binding number}"
                  VerticalAlignment="Top" Width="500" TextAlignment="Left" FontSize="24"
                  FontWeight="Bold" />
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,110,0,0"
                  TextWrapping="Wrap" Text="Took office in:" VerticalAlignment="Top"
                  Width="240" TextAlignment="Right" FontSize="24"
                  Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezInaugurationYear" HorizontalAlignment="Left"
                  Height="32" Margin="250,110,0,0" TextWrapping="Wrap" Text="{Binding
                  inaugurated}" VerticalAlignment="Top" Width="500" TextAlignment="Left"
                  FontSize="24" FontWeight="Bold" />
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,150,0,0"
                  TextWrapping="Wrap" Text="Age at Inauguration:" VerticalAlignment="Top"
                  Width="240" TextAlignment="Right" FontSize="24"
                  Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezInaugurationAge" HorizontalAlignment="Left" Height="32"
                  Margin="250,150,0,0" TextWrapping="Wrap" Text="{Binding inaugurationAge}"
                  VerticalAlignment="Top" Width="500" TextAlignment="Left" FontSize="24"
                  FontWeight="Bold" />
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,190,0,0"
                  TextWrapping="Wrap" Text="Years in Office:" VerticalAlignment="Top"
                  Width="240" TextAlignment="Right" FontSize="24" Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezYearsInOffice" HorizontalAlignment="Left" Height="32"
                  Margin="250,190,0,0" TextWrapping="Wrap" Text="{Binding yearsInOffice}"
```

```
                VerticalAlignment="Top" Width="500" TextAlignment="Left" FontSize="24"
                FontWeight="Bold" />
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,230,0,0"
                TextWrapping="Wrap" Text="Political Party:" VerticalAlignment="Top"
                Width="240" TextAlignment="Right" FontSize="24" Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezParty" HorizontalAlignment="Left" Height="32"
                Margin="250,230,0,0" TextWrapping="Wrap" Text="{Binding politicalParty}"
                VerticalAlignment="Top" Width="500" TextAlignment="Left" FontSize="24"
                FontWeight="Bold" />
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,270,0,0"
                TextWrapping="Wrap" Text="Home State:" VerticalAlignment="Top" Width="240"
                TextAlignment="Right" FontSize="24" Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezHomeState" HorizontalAlignment="Left" Height="32"
                Margin="250,270,0,0" TextWrapping="Wrap" Text="{Binding homeState}"
                VerticalAlignment="Top" Width="500" TextAlignment="Left" FontSize="24"
                FontWeight="Bold" />
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,310,0,0"
                TextWrapping="Wrap" Text="Occupation:" VerticalAlignment="Top" Width="240"
                TextAlignment="Right" FontSize="24" Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezOccupation" HorizontalAlignment="Left" Height="32"
                Margin="250,310,0,0" TextWrapping="Wrap" Text="{Binding occupation}"
                VerticalAlignment="Top" Width="500" TextAlignment="Left" FontSize="24"
                FontWeight="Bold" />
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,350,0,0"
                TextWrapping="Wrap" Text="College:" VerticalAlignment="Top" Width="240"
                TextAlignment="Right" FontSize="24" Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezCollege" HorizontalAlignment="Left" Height="32"
                Margin="250,350,0,0" TextWrapping="Wrap" Text="{Binding college}"
                VerticalAlignment="Top" Width="500" TextAlignment="Left" FontSize="24"
                FontWeight="Bold" />
            <TextBlock HorizontalAlignment="Left" Height="32" Margin="0,390,0,0"
                TextWrapping="Wrap" Text="Historical Rating:" VerticalAlignment="Top"
                Width="240" TextAlignment="Right" FontSize="24" Foreground="#FFBEBCBC"/>
            <TextBlock x:Name="txtPrezRating" Height="32" Margin="250,390,115,0"
                TextWrapping="Wrap" Text="{Binding rating}" VerticalAlignment="Top"
                TextAlignment="Left" FontSize="24" FontWeight="Bold" />
        </Grid>
        <TextBlock x:Name="txtNowShowing" HorizontalAlignment="Left" Height="20"
            Margin="90,710,0,0" TextWrapping="Wrap" Text="Showing All"
            VerticalAlignment="Top" Width="300" Foreground="#FF0BF6DD"/>
        <Image x:Name="imgPrez" Height="500" Width="395" Margin="934,209,37,59"
            Stretch="UniformToFill" />
    </Grid>

    <Page.BottomAppBar>
        <AppBar Background="Purple">
            <Grid>
                <StackPanel Orientation="Horizontal">
                    <Button Style="{StaticResource ViewAllAppBarButtonStyle}"
                        Tapped="ShowAll" />
                    <Button AutomationProperties.Name ="Show Democrats"
                        Style="{StaticResource FilterAppBarButtonStyle}"
                        Tapped="ShowDemocrats" />
                    <Button AutomationProperties.Name ="Show Republicans"
                        Style="{StaticResource FilterAppBarButtonStyle}"
                        Tapped="ShowRepublicans" />
                </StackPanel>
            </Grid>
        </AppBar>
    </Page.BottomAppBar>
</Page>
```

Figure 4 *– An app bar is incorporated for accessing buttons to filter the list of presidents by political party or to display all.*

The styles for the buttons used on the app bar were uncommented in the Common > StandardStyles.xaml file.

Uncommented Styles in the StandardStyles.xaml File

```
<Style x:Key="FilterAppBarButtonStyle" TargetType="ButtonBase" BasedOn="{StaticResource
                                                        AppBarButtonStyle}">
    <Setter Property="AutomationProperties.AutomationId" Value="FilterAppBarButton"/>
    <Setter Property="AutomationProperties.Name" Value="Filter"/>
    <Setter Property="Content" Value="&#xE16E;"/>
</Style>
<Style x:Key="ViewAllAppBarButtonStyle" TargetType="ButtonBase" BasedOn="{StaticResource
                                                      AppBarButtonStyle}">
    <Setter Property="AutomationProperties.AutomationId" Value="ViewAllAppBarButton"/>
    <Setter Property="AutomationProperties.Name" Value="View All"/>
    <Setter Property="Content" Value="&#xE138;"/>
</Style>
```

To utilize JSON data and routines in a project, the JSON package must first be added to the project. From the Project menu, choose “Manage NuGet Packages…”. Select the Json.NET package and click the Install button in the selection’s upper-right corner.

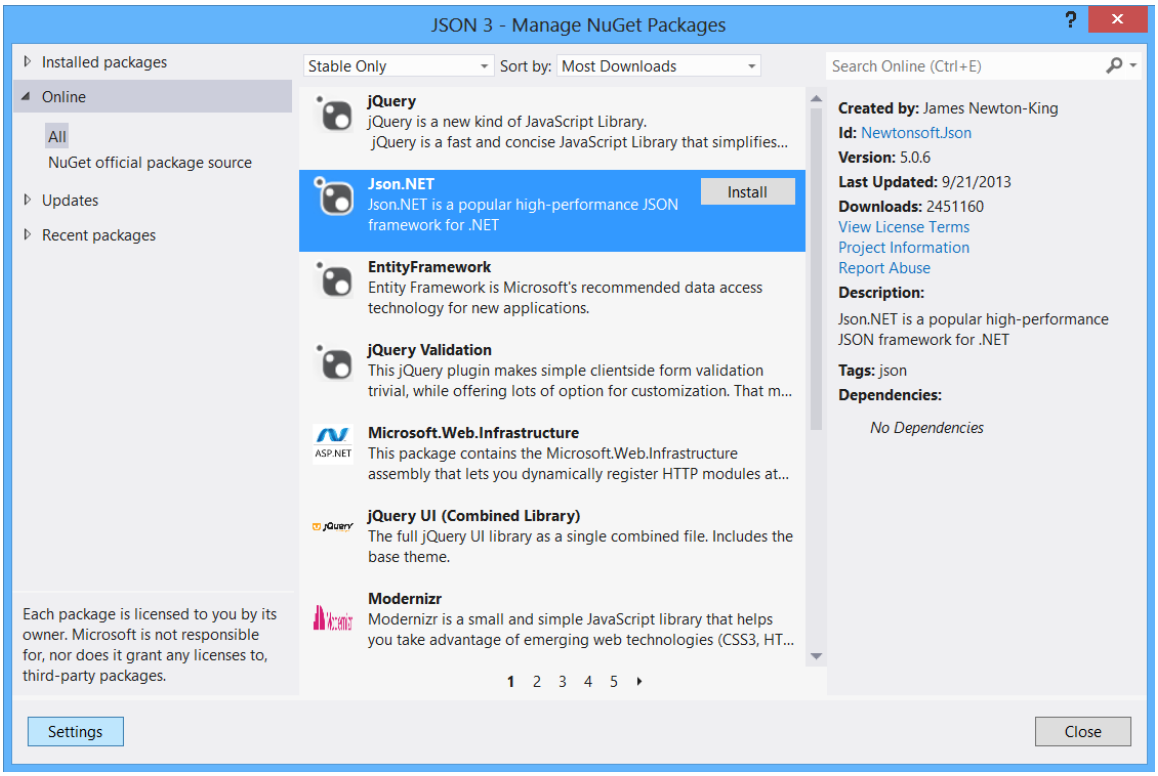

Figure 5 *– Adding the Json.NET package to a project. The dialog is opened from Project > Manage NuGet Packages.*

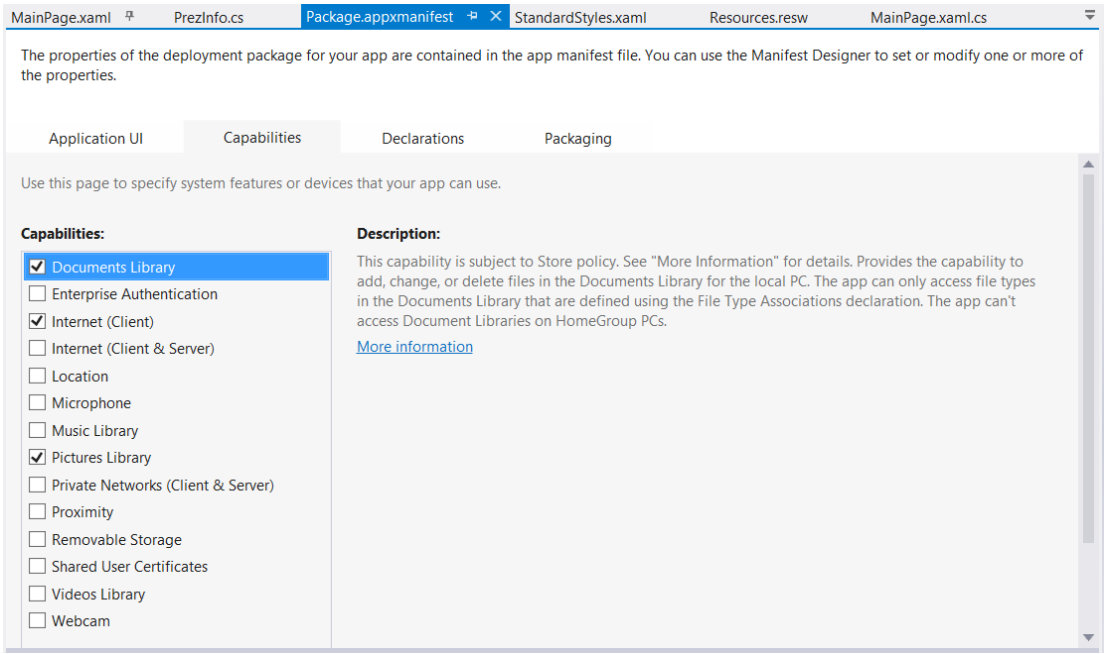

Figure 6 *– Remember to set permissions if the file location (such as in the user's Documents library) requires notification/permission.*

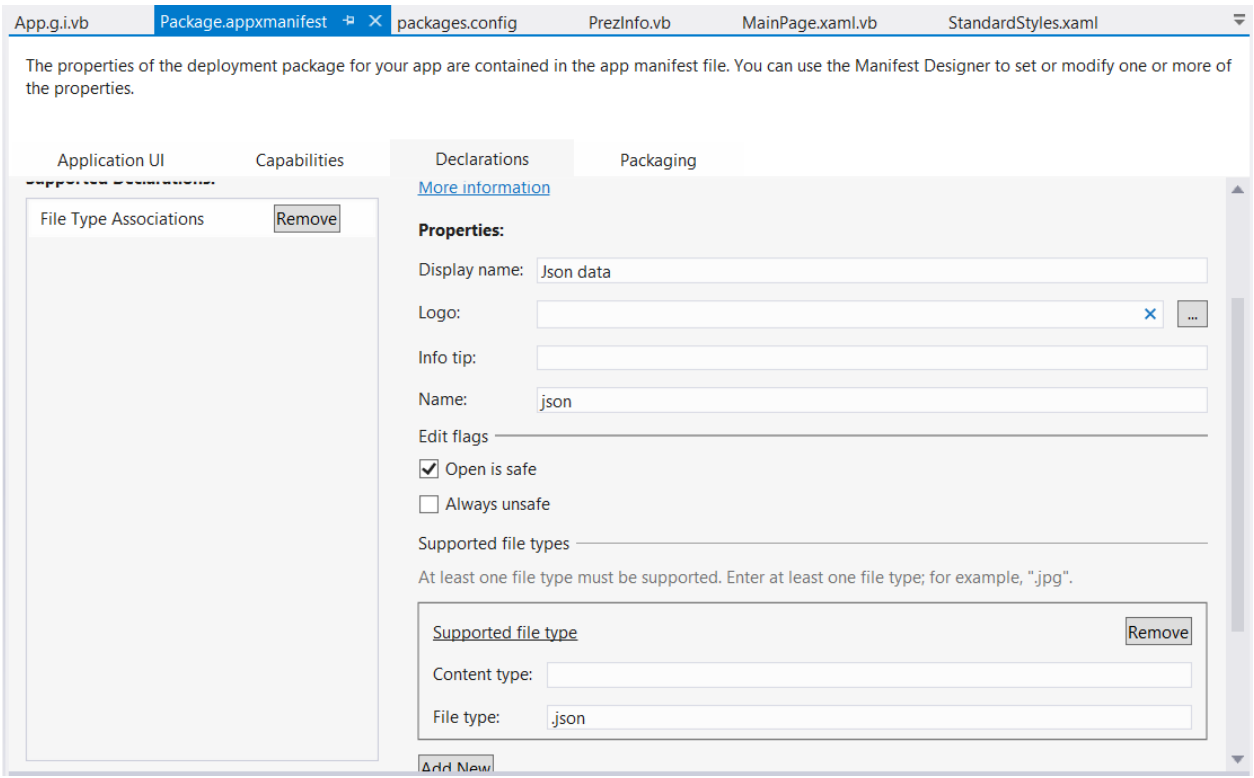

Figure 7 *– The Json file type association is added in the Package.appxmanifest.*

Declarations are established in the *Package.appxmanifest* file to allow for access to the Documents Library and Pictures Library if the data is stored in the User's library. The following code shows how to access a JSON data file in various locations:

- Option 1 (commented out): the User's Documents folder
- Option 2 (commented out): from a String resource embedded in the app (read-only data)
- Option 3: from a web server (read-only data)

Now, you can begin creating the code-behind by adding a new class, named "PrezInfo" (Project menu > Add Class…).

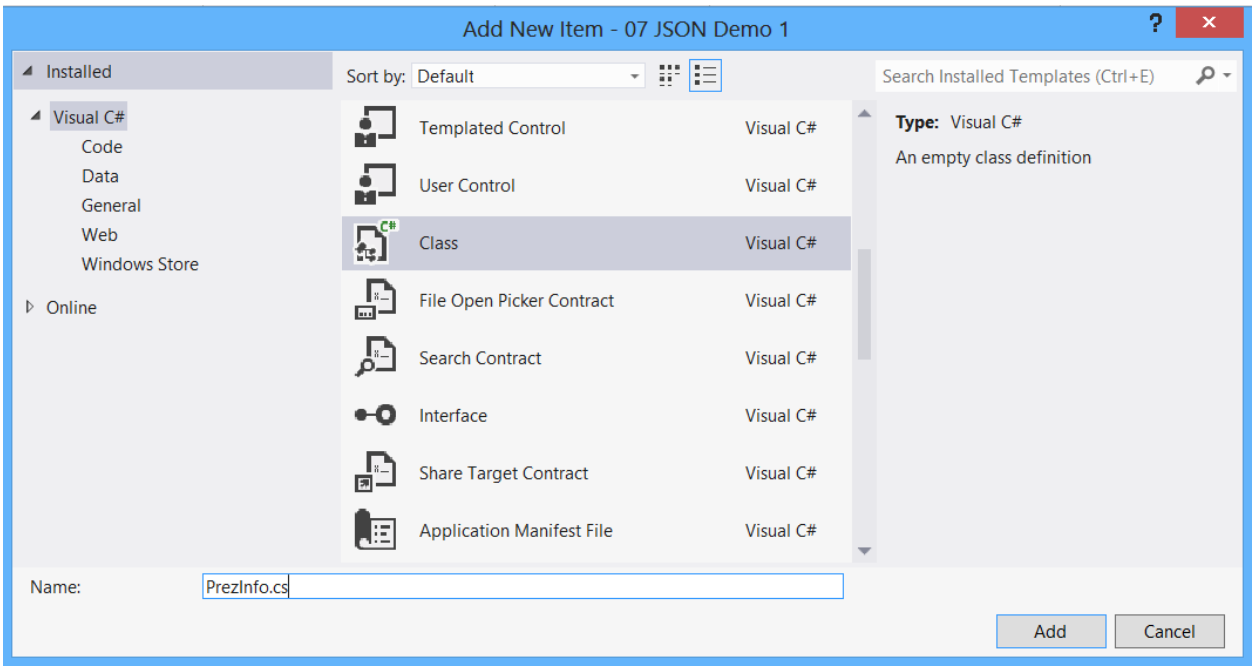

***Figure 8**– The Add New Item dialog for adding a new class to the project.*

The code for the new PrezInfo class establishes properties that correlate to the JSON data records.

C# Code for PrezInfo.cs

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

namespace _07_JSON_Demo_1
{
    class PrezInfo
    {
        public string lastName { get; set; }
        public string firstName { get; set; }
        public string fullName { get; set; } // this is calculated for binding purposes
        public int number { get; set; }
        public int yearsInOffice { get; set; }
        public int inaugurated { get; set; }
        public int inaugurationAge { get; set; }
        public string homeState { get; set; }
        public int rating { get; set; }
        public string politicalParty { get; set; }
        public string occupation { get; set; }
        public string college { get; set; }

        /* SAMPLE JSON RECORD:
           {"lastName":"Bush",
            “firstName":"George H. W. ",
            “number":41,
            “yearsInOffice":4,
```

```
            “inaugurated":1989,
            “inaugurationAge":64,
            “homeState":"Texas",
            “rating":548,
            “politicalParty":"Republican",
            “occupation":"Businessman",
            “college":"Yale"},
        */
    }
}
```

VB Code for PrezInfo.vb Class

```
Public Class PrezInfo
    Public Property lastName As String
    Public Property firstName As String
    Public Property fullName As String 'calculated field for data bound display purposes
    Public Property number As Integer
    Public Property yearsInOffice As Integer
    Public Property inaugurated As Integer
    Public Property inaugurationAge As Integer
    Public Property homeState As String
    Public Property rating As Integer
    Public Property politicalParty As String
    Public Property occupation As String
    Public Property college As String

    ' SAMPLE JSON RECORD:
    ' {"lastName":"Bush",
    ' “firstName":"George H. W. ",
    ' “number":41,
    ' “yearsInOffice":4,
    ' “inaugurated":1989,
    ' “inaugurationAge":64,
    ' “homeState":"Texas",
    ' “rating":548,
    ' “politicalParty":"Republican",
    ' “occupation":"Businessman",
    ' “college":"Yale"},
End Class
```

The code-behind for MainPage establshes a class-level List named presidents to contain the parsed JSON data. Directives are added to provide use of the Newtonsoft.Json, Windows.Storage class, and other classes.

In the OnNavigatedTo event procedure, the JSON data is read from a file into a string named json. The first option (commented out) reads the data from a file in the user's Documents library. The second option, also commented out, shows how to read JSON data that is embedded as a String resourse in the app's source files. The third option reads the data from a web server, and this code is uncommented and is used in this project. (The images displayed also are read from the server.)

Here is the beauty and simplicity of working with JSON data. Once the data is copied to a string, the string is de-serialized into a List of type PrezInfo and populated into the President List. The DeserializeObject method is called from the JsonConvert class. Here is an example: presidents = JsonConvert.DeserializeObject<List<PrezInfo>>(json);

A for each loop calculates the value of the fullname member for each record in the presidents List. This value is used for binding each record listing in the ListView of the MainPage. The ItemSource for the lvpresidents LIstView is established and the data from the first record (element [0]) is displayed.

The ShowImage() method called to display the chosen president's photograph is shown with two options of code. Option 1 contains an example of opening the image from a "Presidents" folder within the user's Pictures Library. Option 2 (used in the example) accesses the images from a webserver that is located within a "Presidents" folder as well.

The ListView responds to an ItemClicked event that is handled by the PresidentChosen( ) event method. All the record fields (TextBlock controls) are located within a Grid control, and the data source (DataContext) of the grid (grdTest) is set to the particular record. Each of the TextBlock controls are bound to the data source.The SHowImage() method is called to update the president's photograph accordingly.

Finally, the code-behind contains three methods to handle the three App Bar buttons to filter the ListView items to show all the presidents, show only the Democrats, and show only the Repubicans.

C# Code for MainPage.xaml.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using Windows.Foundation;
using Windows.Foundation.Collections;
using Windows.UI.Xaml;
using Windows.UI.Xaml.Controls;
using Windows.UI.Xaml.Controls.Primitives;
using Windows.UI.Xaml.Data;
using Windows.UI.Xaml.Input;
using Windows.UI.Xaml.Media;
using Windows.UI.Xaml.Navigation;

using Newtonsoft.Json;
using Newtonsoft.Json.Utilities;
using Windows.Storage;
using Windows.ApplicationModel.Resources;
using Windows.UI.Xaml.Media.Imaging;
using Windows.Storage.Streams;
using System.Net.Http;


namespace _07_JSON_Demo_1
{
    public sealed partial class MainPage : Page
    {
        List<PrezInfo> presidents = new List<PrezInfo>();
        PrezInfo chosenPrez;
        StorageFolder myPicsFolder = KnownFolders.PicturesLibrary;

        public MainPage()
        {
```

```
            this.InitializeComponent();
        }

        protected async override void OnNavigatedTo(NavigationEventArgs e)
        {
            //OPTION 1 : read and write data to the User's Dcouments folder
            /*StorageFolder myAppFolder = KnownFolders.DocumentsLibrary;
            string dataFileName = "presidents.json";
            StorageFile myData = await myAppFolder.GetFileAsync(dataFileName);
            StreamReader sr = new StreamReader(await myData.OpenStreamForReadAsync());
            string json = await sr.ReadToEndAsync(); */


            //OPTION 2 - read data from internal String Resource (used if data is static)
            /*ResourceLoader rl = new ResourceLoader("Resources");
            string json = rl.GetString("String1"); */


            //OPTION 3 - Reading JSON data from a website
            //add: using System.Net.Http;
            Uri mySite = new Uri("http://www.tricalico.com/presidents.json");
            HttpRequestMessage request = new HttpRequestMessage(HttpMethod.Get, mySite);
            HttpClient client = new HttpClient();
            HttpResponseMessage response = await client.SendAsync(request,
                                        HttpCompletionOption.ResponseHeadersRead);
            string json =  await response.Content.ReadAsStringAsync();
            // end option 4

            if (!string.IsNullOrEmpty(json))
            {
                presidents = JsonConvert.DeserializeObject<List<PrezInfo>>(json);
            }
            foreach (PrezInfo pi in presidents)
            { //calculate the fullName value for each president for Binding use
                pi.fullName = pi.firstName + " " + pi.lastName;
            }
            //sr.Dispose();   //uncomment this line for option 1
            lvPresidents.ItemsSource = presidents;
            chosenPrez = (PrezInfo) lvPresidents.Items[0];
            gridTest.DataContext = chosenPrez;
            ShowImage(chosenPrez.lastName, chosenPrez.firstName);
        }

        private void PresidentChosen(object sender, ItemClickEventArgs e)
        {
            chosenPrez = (PrezInfo) e.ClickedItem;
            gridTest.DataContext = chosenPrez;
            ShowImage(chosenPrez.lastName, chosenPrez.firstName);
        }

        private void ShowImage(string ln, string fn)
        {
            //OPTION 1 Images stored locally
            /*
            StorageFolder myPicsSubFolder = await
                                myPicsFolder.GetFolderAsync("Presidents");
            // note: Add 'async' to the method declaration.
            string imageFile = (ln + "_" + fn.Substring(0, 1) + ".png").ToLower();
```

```
            StorageFile file = await myPicsSubFolder.GetFileAsync(imageFile);
            BitmapImage bitmap = new BitmapImage();  //create a new bitmap
            // Open a filestream to the bitmap
            // add using Windows.Storage.Streams;
            IRandomAccessStream fileStream = await
                                file.OpenAsync(Windows.Storage.FileAccessMode.Read);
            bitmap.SetSource(fileStream);
            //set the source for the Image XAML control
            imgPrez.Source = bitmap;
             */

            //OPTION 2: Images stored on server
            //NOTE: Remove async from method declaration
            string imageFile = (ln + "_" + fn.Substring(0, 1) + ".png").ToLower();
            Uri mySite = new Uri("http://www.tricalico.com/Presidents/" + imageFile);
            imgPrez.Source = new BitmapImage(mySite);
        }

        private void ShowAll(object sender, TappedRoutedEventArgs e)
        {
            lvPresidents.ItemsSource = presidents;
            chosenPrez = (PrezInfo)lvPresidents.Items[0];
            gridTest.DataContext = chosenPrez;
            ShowImage(chosenPrez.lastName, chosenPrez.firstName);
            txtNowShowing.Text = "Showing ALL";
        }

        private void ShowDemocrats(object sender, TappedRoutedEventArgs e)
        {   //Use LINQ to filter
            var dems = from value in presidents
                       where value.politicalParty.Equals("Democrat")
                       select value;
            lvPresidents.ItemsSource = dems;
            chosenPrez = (PrezInfo)lvPresidents.Items[0];
            gridTest.DataContext = chosenPrez;
            ShowImage(chosenPrez.lastName, chosenPrez.firstName);
            txtNowShowing.Text = "Showing DEMOCRATS only";
        }


        private void ShowRepublicans(object sender, TappedRoutedEventArgs e)
        {   //Use LINQ to filter
            var reps = from value in presidents
                       where value.politicalParty.Equals("Republican")
                       select value;
            lvPresidents.ItemsSource = reps;
            chosenPrez = (PrezInfo)lvPresidents.Items[0]; ;
            gridTest.DataContext = chosenPrez;
            ShowImage(chosenPrez.lastName, chosenPrez.firstName);
            txtNowShowing.Text = "Showing REPUBLICANS only";
        }
    }
}
```

VB.NET Code Behind for MainPage.xaml.vb

```
Imports Newtonsoft.Json
Imports Newtonsoft.Json.Utilities
```

```
Imports Windows.Storage
Imports Windows.ApplicationModel.Resources
Imports Windows.UI.Xaml.Media.Imaging
Imports Windows.Storage.Streams
Imports System.Net.Http

Public NotInheritable Class MainPage
    Inherits Page

    Dim presidents As List(Of PrezInfo)
    Dim chosenPrez As PrezInfo
    Dim myPicsFolder As StorageFolder = KnownFolders.PicturesLibrary


    Protected Overrides Async Sub OnNavigatedTo(e As Navigation.NavigationEventArgs)
        ''OPTION 1: ead and write data to the User's Dcouments folder
        'Dim myAppFolder As StorageFolder = KnownFolders.DocumentsLibrary
        'Dim dataFileName As String = "presidents.json"
        'Dim myData As StorageFile = Await myAppFolder.GetFileAsync(dataFileName)
        'Dim sr As StreamReader = New StreamReader(Await myData.OpenStreamForReadAsync())
        'Dim json As String = Await sr.ReadToEndAsync()

        ''OPTION 2 - read data from internal String Resource (used if data is static)
        'Dim rl As ResourceLoader = New ResourceLoader("Resources")
        'Dim json As String = rl.GetString("String1")

        'OPTION 3 - Reading JSON data from a website
        ' add:    Using System.Net.Http
        Dim mySite As Uri = New Uri("http://www.tricalico.com/presidents.json")
        Dim request As HttpRequestMessage = New HttpRequestMessage(HttpMethod.Get, mySite)
        Dim client As HttpClient = New HttpClient()
        Dim response As HttpResponseMessage = Await client.SendAsync(request, _
                                    HttpCompletionOption.ResponseHeadersRead)
        Dim json As String = Await response.Content.ReadAsStringAsync()
        'end option 4

        If (Not String.IsNullOrEmpty(json)) Then
            presidents = JsonConvert.DeserializeObject(Of List(Of PrezInfo))(json)
        End If

        For Each pi As PrezInfo In presidents
            ' calculate the fullName value for each president for Binding use
            pi.fullName = pi.firstName + " " + pi.lastName
        Next
        ' sr.Dispose()  'uncomment this line for option 1
        lvPresidents.ItemsSource = presidents
        chosenPrez = lvPresidents.Items(0)
        gridTest.DataContext = chosenPrez
        ShowImage(chosenPrez.lastName, chosenPrez.firstName)

    End Sub

    Private Sub ShowImage(ln As String, fn As String)
        '' OPTION 1 Images stored locally
        'Dim myPicsSubFolder As StorageFolder = Await _
        '                        myPicsFolder.GetFolderAsync("Presidents")
        ''note: Add ‘async’ to the method declaration.
        'Dim imageFile As String = (ln & "_" & fn.Substring(0, 1) & ".png").ToLower()

        'Dim file As StorageFile = Await myPicsSubFolder.GetFileAsync(imageFile)
        'Dim bitmap As BitmapImage = New BitmapImage()  'create a new bitmap
```

```
        ''Open a filestream to the bitmap
        '' add using Windows.Storage.Streams;
        'Dim fileStream As IRandomAccessStream = Await _
        '                       file.OpenAsync(Windows.Storage.FileAccessMode.Read)
        'bitmap.SetSource(fileStream)
        ''set the source for the Image XAML control
        'imgPrez.Source = bitmap

        'OPTION 2: Images stored on server
        'NOTE: Remove async from method declaration
        Dim imageFile As String = (ln & "_" & fn.Substring(0, 1) & ".png").ToLower()
        Dim mySite As Uri = New Uri("http://www.tricalico.com/Presidents/" + imageFile)
        imgPrez.Source = New BitmapImage(mySite)

    End Sub

    Private Sub PresidentChosen(sender As Object, e As ItemClickEventArgs) _
                                                    Handles lvPresidents.ItemClick
        chosenPrez = e.ClickedItem
        gridTest.DataContext = chosenPrez
        ShowImage(chosenPrez.lastName, chosenPrez.firstName)
    End Sub

    Private Sub ShowAll(sender As Object, e As TappedRoutedEventArgs)
        lvPresidents.ItemsSource = presidents
        chosenPrez = lvPresidents.Items(0)
        gridTest.DataContext = chosenPrez
        ShowImage(chosenPrez.lastName, chosenPrez.firstName)
        txtNowShowing.Text = "Showing ALL"
    End Sub

    Private Sub ShowDemocrats(sender As Object, e As TappedRoutedEventArgs)
        ' Use LINQ to filter
        Dim dems = From value In presidents
                   Where value.politicalParty.Equals("Democrat")
                   Select value
        lvPresidents.ItemsSource = dems
        chosenPrez = lvPresidents.Items(0)
        gridTest.DataContext = chosenPrez
        ShowImage(chosenPrez.lastName, chosenPrez.firstName)
        txtNowShowing.Text = "Showing DEMOCRATS only"
    End Sub

    Private Sub ShowRepublicans(sender As Object, e As TappedRoutedEventArgs)
        ' Use LINQ to filter
        Dim reps = From value In presidents
                   Where value.politicalParty.Equals("Republican")
                   Select value
        lvPresidents.ItemsSource = reps
        chosenPrez = lvPresidents.Items(0)
        gridTest.DataContext = chosenPrez
        ShowImage(chosenPrez.lastName, chosenPrez.firstName)
        txtNowShowing.Text = "Showing REPUBLICANS only"
    End Sub
End Class
```

Writing Data to a JSON Formatted File

You used the DeserializeObject( ) method of the JsonConvert class to parse a JSON-formatted string into a List of objects. Likewise, there is a SerializeObject() method that

functions to convert an object to a JSON-formatted string. You can use a StreamWriter to add records to a JSON data file. In the following example, you will use this technique to create an interactive address book in which the user can view the JSON data and add new records to the data file.

In this project, you will read and write data to the user's roaming settings so that the data can be shared among multiple devices.

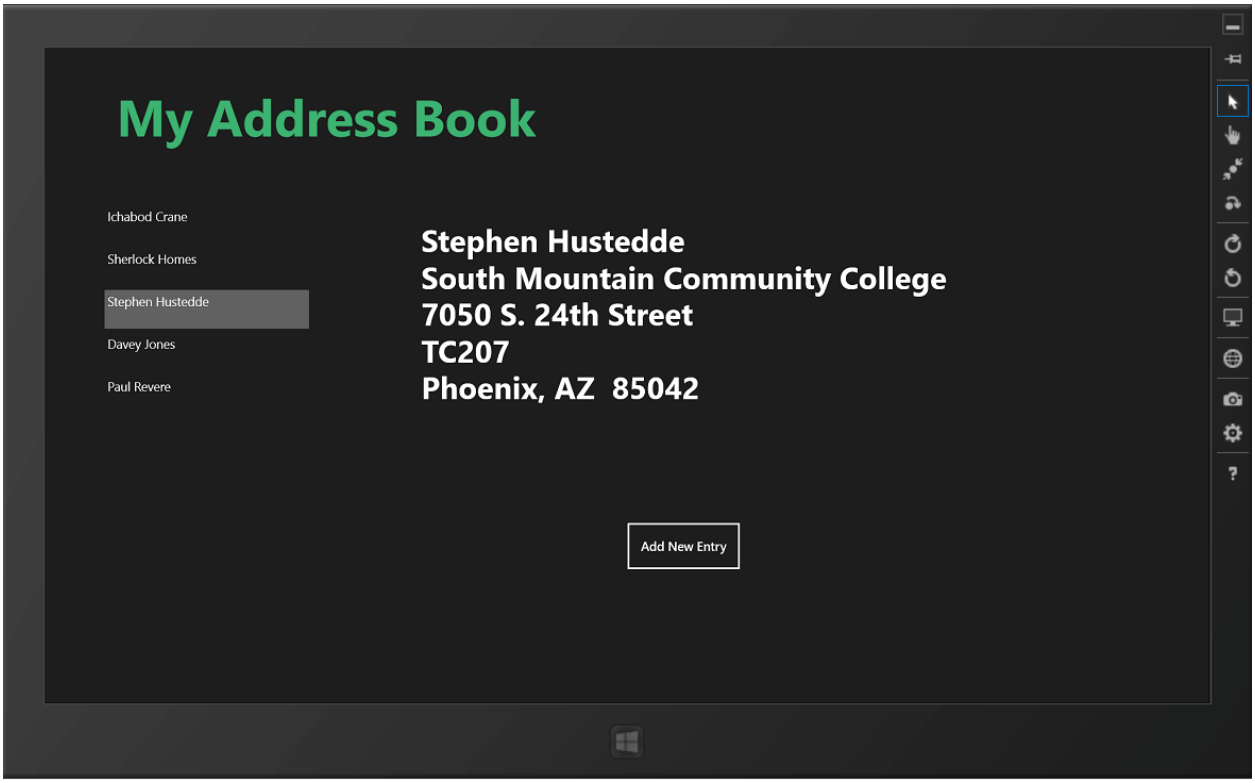

Figure 9 *– The My Address Book app reads data from a JSON file to populate a ListView of names on the left and displays the data from the user's selection in a TextBlock. A DeserializeObject method is employed to convert the JSON string data to a list of custom class objects.*

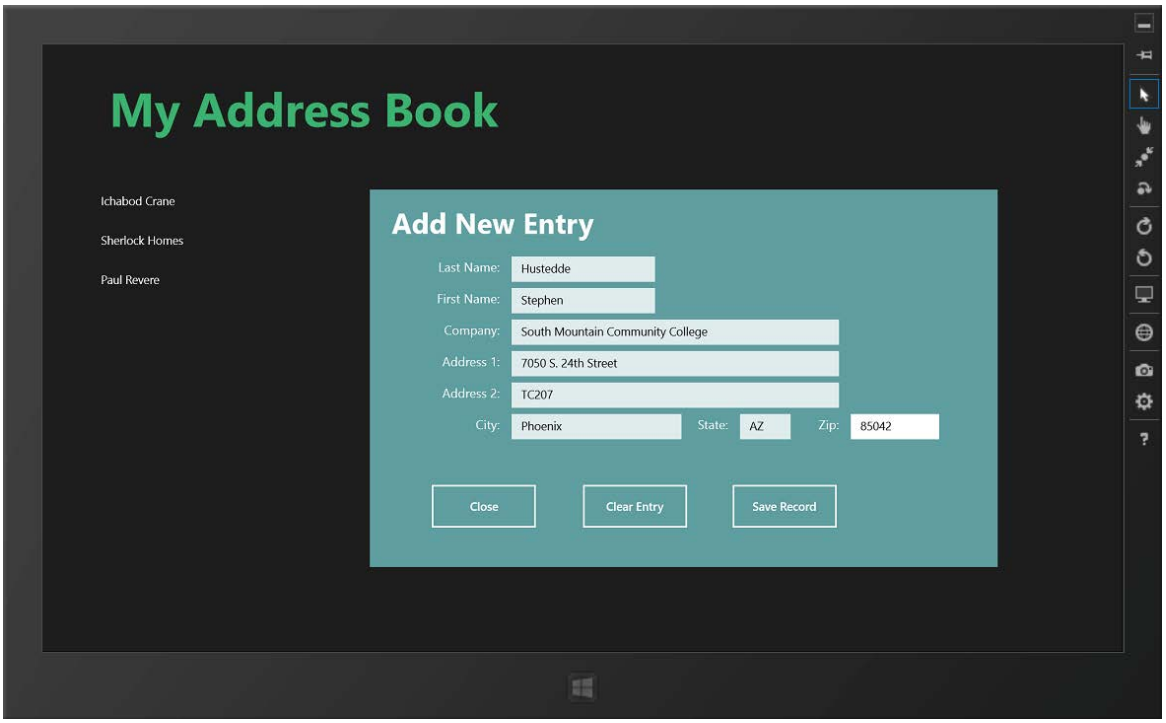

Figure 10 *– A grid containing TextBoxes and Buttons is displayed for the user to add a new record to the data file. The underlying code uses a JSON SerializeObject method to convert the data to a JSON string.*

The project was created using a Blank template. The XAML code defining the MainPage.xaml document utilizes a ListView and a TextBlock to display all the records and the record details, respectively. A button displays a collapsed grid to add a new record. That collapsed grid contains numerous TextBox controls to accept input from the user and buttons to clear the textboxes, close the grid, and save the data.

XAML Code for MainPage.xaml

```
<Page
    x:Class="_07_Roaming_Data___Address_Book_2.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:_07_Roaming_Data___Address_Book_2"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">

    <Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}">
        <TextBlock HorizontalAlignment="Left" Height="86" Margin="85,50,0,0"
              TextWrapping="Wrap" Text="My Address Book" VerticalAlignment="Top"
              Width="1214" Foreground="MediumSeaGreen" FontSize="60" FontWeight="Bold"/>
        <Grid x:Name="grdData"  HorizontalAlignment="Left" Height="479"
              VerticalAlignment="Top" Width="940" Margin="416,184,0,0">
            <TextBlock x:Name="txtAddress" HorizontalAlignment="Left" TextWrapping="Wrap"
                  Text="Choose a name at the left" VerticalAlignment="Top"
                  Margin="28,24,0,0" FontSize="36" FontWeight="Bold" Height="343"
                  Width="902"/>
            <Button Content="Add New Entry" HorizontalAlignment="Left" Height="60"
                  Margin="268,372,0,0" VerticalAlignment="Top" Width="138"
                  Click="ShowAddNewEntry"  />
```

```
        </Grid>

        <Grid x:Name="grdAdd" Visibility="Collapsed" Background="CadetBlue"
              HorizontalAlignment="Left" Height="479" VerticalAlignment="Top" Width="797"
              Margin="416,184,0,0">
            <TextBlock HorizontalAlignment="Left" TextWrapping="Wrap" Text="Add New
                  Entry" VerticalAlignment="Top" Margin="28,24,0,0" FontSize="36"
                  FontWeight="Bold"/>
            <TextBlock HorizontalAlignment="Left" Height="23" Margin="53,90,0,0"
                  TextWrapping="Wrap" Text="Last Name:" VerticalAlignment="Top"
                  Width="112" FontSize="16" TextAlignment="Right"/>
            <TextBox x:Name="txtAddLastName" HorizontalAlignment="Left"
                  TextWrapping="Wrap" Text="" VerticalAlignment="Top"
                  RenderTransformOrigin="2.411,3.229" Margin="180,85,0,0" Width="182"/>
            <TextBlock HorizontalAlignment="Left" Height="23" Margin="53,130,0,0"
                  TextWrapping="Wrap" Text="First Name:" VerticalAlignment="Top"
                  Width="112" FontSize="16" TextAlignment="Right"/>
            <TextBox x:Name="txtAddFirstName" HorizontalAlignment="Left"
                  TextWrapping="Wrap" Text="" VerticalAlignment="Top"
                  RenderTransformOrigin="2.411,3.229" Margin="180,125,0,0" Width="182"/>
            <TextBlock HorizontalAlignment="Left" Height="23" Margin="53,170,0,0"
                  TextWrapping="Wrap" Text="Company:" VerticalAlignment="Top" Width="112"
                  FontSize="16" TextAlignment="Right"/>
            <TextBox x:Name="txtAddCompany" HorizontalAlignment="Left"
                  TextWrapping="Wrap" Text="" VerticalAlignment="Top"
                  RenderTransformOrigin="2.411,3.229" Margin="180,165,0,0" Width="416"/>
            <TextBlock HorizontalAlignment="Left" Height="23" Margin="53,210,0,0"
                  TextWrapping="Wrap" Text="Address 1:" VerticalAlignment="Top"
                  Width="112" FontSize="16" TextAlignment="Right"/>
            <TextBox x:Name="txtAddAddress1" HorizontalAlignment="Left"
                  TextWrapping="Wrap" Text="" VerticalAlignment="Top"
                  RenderTransformOrigin="2.411,3.229" Margin="180,205,0,0" Width="416"/>
            <TextBlock HorizontalAlignment="Left" Height="23" Margin="53,250,0,0"
                  TextWrapping="Wrap" Text="Address 2:" VerticalAlignment="Top"
                  Width="112" FontSize="16" TextAlignment="Right"/>
            <TextBox x:Name="txtAddAddress2" HorizontalAlignment="Left"
                  TextWrapping="Wrap" Text="" VerticalAlignment="Top"
                  RenderTransformOrigin="2.411,3.229" Margin="180,245,0,0" Width="416"/>
            <TextBlock HorizontalAlignment="Left" Height="23" Margin="53,290,0,0"
                  TextWrapping="Wrap" Text="City:" VerticalAlignment="Top" Width="112"
                  FontSize="16" TextAlignment="Right"/>
            <TextBox x:Name="txtAddCity" HorizontalAlignment="Left" TextWrapping="Wrap"
                  Text="" VerticalAlignment="Top" RenderTransformOrigin="2.411,3.229"
                  Margin="180,285,0,0" Width="216"/>
            <TextBlock HorizontalAlignment="Left" Height="27" Margin="407,290,0,0"
                  TextWrapping="Wrap" Text="State:" VerticalAlignment="Top" Width="48"
                  FontSize="16" TextAlignment="Right"/>
            <TextBox x:Name="txtAddState" HorizontalAlignment="Left" TextWrapping="Wrap"
                  Text="" VerticalAlignment="Top" RenderTransformOrigin="2.411,3.229"
                  Margin="470,285,0,0" Width="29"/>
            <TextBlock HorizontalAlignment="Left" Height="27" Margin="548,290,0,0"
                  TextWrapping="Wrap" Text="Zip:" VerticalAlignment="Top" Width="48"
                  FontSize="16" TextAlignment="Right"/>
            <TextBox x:Name="txtAddZip" HorizontalAlignment="Left" TextWrapping="Wrap"
                  Text="" VerticalAlignment="Top" RenderTransformOrigin="2.411,3.229"
                  Margin="611,285,0,0" Width="112"/>
            <Button Content="Clear Entry" HorizontalAlignment="Left" Height="60"
                  Margin="268,372,0,0" VerticalAlignment="Top" Width="138"
```

```
                Click="ClearEntry"/>
            <Button Content="Save Record" HorizontalAlignment="Left" Height="60"
                Margin="458,372,0,0" VerticalAlignment="Top" Width="138"
                Click="SaveRecord"/>
            <Button Content="Close" HorizontalAlignment="Left" Height="60"
                Margin="76,372,0,0" VerticalAlignment="Top" Width="138"
                Click="CloseAdd"/>
        </Grid>
        <ListView x:Name="lvList" HorizontalAlignment="Left" Height="390"
            Margin="70,184,0,0" VerticalAlignment="Top" Width="261"
            ItemClick="ShowAddress" IsItemClickEnabled="True" />
    </Grid>
</Page>
```

As seen in the previous project, the Json.NET package must be added to the project to be able to work with the JSON data. This is installed via the Project menu > Manage NuGet Packages… menu item. (See Figure 5 in the Lesson 7 Guide.)

In the following code behind for the MainPage.xaml page, an AddressInfo custom class was created at the end of the code. This could have been created as a new Class document as was done for the PrezInfo class in the previous project. This class defines the properties for an address object.

A class level List containing AddressInfo objects is created to hold the data. The ApplicationDataContainer and StorageFolder are established as being in the user's roaming settings, and a file name of "myAddressBook.json" is established.

When the app is opened, the OnNavigatedTo( ) event method calls the SetDataFile() methods. This process creates a blank "myAddressBook.json" file in the user's roaming account if it does not already exist. Next, the event method calls the LoadDate( ) method, which retrieves the JSON string and parses it into the List of AddressInfo objects. It also sorts the objects by last name and secondarily by first name, and then adds each name to the ListView of this page. From the ListView, the user can choose to display any selected record. The parsing of the json string is accomplished with a JsonConvert.DeserializeObject() method as was explained in the previous project.

Clicking on an item in the ListView triggers the ShowAddress( ) method. It simply builds a multiline string from the data record info and then displays it in a TextBlock.

A click or tap on the "Add New Record" button executes the ShowAddNewEntry( ) method, which simply makes the collapsed "grdAdd" grid visible. Users can then enter data into the various textbox fields. The Close button collapses the grid again, while the Clear Entry button sets all the textboxes of grid to an empty string.

The "Save Record" button calls the SaveRecord method. It creates a new AddressInfo object and sets its properties to the text of the various textboxes—before serializing the object into a JSON string—using the JsonConvert.SerializeObject( ) method to accomplish the conversion. An asynchronous streamwriter writes the data to the roaming file and then reloads the data to refresh the ListView.

C# Code for MainPage.xaml.cs

```
using System;
using System.Collections.Generic;
using System.IO;
```

```
using System.Linq;
using Windows.Foundation;
using Windows.Foundation.Collections;
using Windows.UI.Xaml;
using Windows.UI.Xaml.Controls;
using Windows.UI.Xaml.Controls.Primitives;
using Windows.UI.Xaml.Data;
using Windows.UI.Xaml.Input;
using Windows.UI.Xaml.Media;
using Windows.UI.Xaml.Navigation;
using Windows.Storage;
using Newtonsoft.Json;
using Newtonsoft.Json.Serialization;


namespace _07_Roaming_Data___Address_Book_2
{

    public sealed partial class MainPage : Page
    {
        List<AddressInfo> addressBook = new List<AddressInfo>();
        // Add: using Windows.Storage;
        ApplicationDataContainer myAppData = ApplicationData.Current.RoamingSettings;
        //read and write data to the User's Roaming settings
        StorageFolder myAppFolder = ApplicationData.Current.RoamingFolder;
        string dataFileName = "myAddressBook.json";

        public MainPage()
        {
            this.InitializeComponent();

        }

        protected override void OnNavigatedTo(NavigationEventArgs e)
        {
            SetDataFile();
            LoadData();
        }

        private async void SetDataFile()
        {
            try
            {
                //create the data file if it doesn't exist
                await myAppFolder.CreateFileAsync(dataFileName);
            }
            catch
            {
            }
        }

        private async void LoadData()
        {
            StorageFile myData = await myAppFolder.GetFileAsync(dataFileName);
            StreamReader sr = new StreamReader(await myData.OpenStreamForReadAsync());
            // Choose Project > Manage Nuget Packages...   then install JSON
            // Add: using Newtonsoft.Json;
            // Add:  using Newtonsoft.Json.Serialization;

            string json = await sr.ReadToEndAsync();
            lvList.Items.Clear();
```

```
            if (!string.IsNullOrEmpty(json))
            {
                addressBook = JsonConvert.DeserializeObject<List<AddressInfo>>(json);
            }
            var sortedAddresses = from value in addressBook
                        orderby value.lastName + value.firstName
                        select value;
            foreach (AddressInfo ai in sortedAddresses)
            {
                lvList.Items.Add(ai.fullName);
            }
            sr.Dispose();
        }

        private void ShowAddress(object sender, ItemClickEventArgs e)
        {
            string fn = e.ClickedItem.ToString();
            //AddressInfo xyz = addressBook.IndexOf( e.ClickedItem.ToString);
            var  addresses = from value in addressBook
                       where value.fullName.Equals(fn)
                       select value;
            string abc = "";
            foreach (AddressInfo xyz in addresses)  // generally should only return 1
            {
                abc += xyz.fullName;
                if (xyz.company != "")
                {
                    abc += "\n" + xyz.company;
                }
                abc += "\n" + xyz.address1;
                if (xyz.address2 != "")
                {
                    abc += "\n" + xyz.address2;
                }
                abc += "\n" + xyz.city + ", " + xyz.state + "  " + xyz.zip + "\n\n";
            }
            txtAddress.Text = abc;
        }

        private void ShowAddNewEntry(object sender, RoutedEventArgs e)
        {
            grdAdd.Visibility = Windows.UI.Xaml.Visibility.Visible;
        }

        private void CloseAdd(object sender, RoutedEventArgs e)
        {
            grdAdd.Visibility = Visibility.Collapsed;
        }

        private void ClearEntry(object sender, RoutedEventArgs e)
        {
            ClearAddScreen();
        }

        private void ClearAddScreen()
        {
            txtAddLastName.Text = String.Empty;
            txtAddFirstName.Text = String.Empty;
            txtAddCompany.Text = String.Empty;
            txtAddAddress1.Text = String.Empty;
            txtAddAddress2.Text = String.Empty;
```

```
            txtAddCity.Text = String.Empty;
            txtAddState.Text = String.Empty;
            txtAddZip.Text = String.Empty;
            txtAddLastName.Focus(Windows.UI.Xaml.FocusState.Keyboard);
        }


        private async void SaveRecord(object sender, RoutedEventArgs e)
        {
            AddressInfo newAddress = new AddressInfo();
            newAddress.lastName = txtAddLastName.Text;
            newAddress.firstName = txtAddFirstName.Text;
            newAddress.fullName = txtAddFirstName.Text + " " + txtAddLastName.Text;
            newAddress.company = txtAddCompany.Text;
            newAddress.address1 = txtAddAddress1.Text;
            newAddress.address2 = txtAddAddress2.Text;
            newAddress.city = txtAddCity.Text;
            newAddress.state = txtAddState.Text;
            newAddress.zip = txtAddZip.Text;
            addressBook.Add(newAddress);
            // Choose Project > Manage Nuget Packages...   then install JSON
            // Add: using Newtonsoft.Json;
            // Add:  using Newtonsoft.Json.Serialization;
            var json = JsonConvert.SerializeObject(addressBook);
            System.IO.Stream myStream = await
                            myAppFolder.OpenStreamForWriteAsync(dataFileName,
                            CreationCollisionOption.ReplaceExisting);
            StreamWriter sw = new StreamWriter(myStream);
            await sw.WriteAsync(json);
            sw.Dispose();
            ClearAddScreen();
            LoadData();
        }
    }

    public class AddressInfo
    {
        public string lastName { get; set; }
        public string firstName { get; set; }
        public string fullName { get; set; }
        public string company { get; set; }
        public string address1 { get; set; }
        public string address2 { get; set; }
        public string city { get; set; }
        public string state { get; set; }
        public string zip { get; set; }
    }
}
```

VB.NET Code for MainPage.xaml.vb

```
Imports Windows.Storage
Imports Newtonsoft.Json
Imports Newtonsoft.Json.Serialization
Imports System.Linq
Imports System
Imports System.Collections.Generic
Imports System.IO

Public NotInheritable Class MainPage
    Inherits Page
```

```
Dim addressBook As List(Of AddressInfo) = New List(Of AddressInfo)()
' Add: using Windows.Storage
Dim myAppData As ApplicationDataContainer = ApplicationData.Current.RoamingSettings
'read and write data to the User's Roaming settings
Dim myAppFolder As StorageFolder = ApplicationData.Current.RoamingFolder
Dim dataFileName As String = "myAddressBook.json"

Protected Overrides Sub OnNavigatedTo(e As Navigation.NavigationEventArgs)
    SetDataFile()
    LoadData()
End Sub

Private Async Sub SetDataFile()
    Try
        'create the data file if it doesn't exist
        Await myAppFolder.CreateFileAsync(dataFileName)
    Catch ex As Exception

    End Try
End Sub

Private Async Sub LoadData()
    Dim myData As StorageFile = Await myAppFolder.GetFileAsync(dataFileName)
    Dim sr As StreamReader = New StreamReader(Await myData.OpenStreamForReadAsync())
    ' Choose Project > Manage Nuget Packages...   then install JSON
    ' Add: using Newtonsoft.Json
    ' Add:  using Newtonsoft.Json.Serialization
    Dim json As String = Await sr.ReadToEndAsync()
    lvList.Items.Clear()
    If (Not String.IsNullOrEmpty(json)) Then
        addressBook = JsonConvert.DeserializeObject(Of List(Of AddressInfo))(json)
        Dim sortedAddresses = From value In addressBook
                              Select value
                              Order By value.lastName & value.firstName

        For Each ai As AddressInfo In sortedAddresses
            lvList.Items.Add(ai.fullName)
        Next
    End If
    sr.Dispose()
End Sub

Private Sub ShowAddress(sender As Object, e As ItemClickEventArgs) _
                                                Handles lvList.ItemClick
    Dim fn As String = e.ClickedItem.ToString()
    Dim addresses = From value In addressBook
                Where value.fullName.Equals(fn)
                Select value

    Dim abc As String = ""

    For Each xyz As AddressInfo In addresses  'generally should only return 1
        abc &= xyz.fullName
        If (xyz.company <> "") Then
            abc &= vbCrLf & xyz.company
        End If
        abc &= vbCrLf & xyz.address1
        If (xyz.address2 <> "") Then
            abc &= vbCrLf & xyz.address2
        End If
```

```
            abc &= vbCrLf & xyz.city & ", " & xyz.state & "  " & xyz.zip & vbCrLf & vbCrLf
        Next
        txtAddress.Text = abc
    End Sub

    Private Sub ClearEntry(sender As Object, e As RoutedEventArgs)
        ClearAddScreen()
    End Sub

    Private Sub ClearAddScreen()
        txtAddLastName.Text = String.Empty
        txtAddFirstName.Text = String.Empty
        txtAddCompany.Text = String.Empty
        txtAddAddress1.Text = String.Empty
        txtAddAddress2.Text = String.Empty
        txtAddCity.Text = String.Empty
        txtAddState.Text = String.Empty
        txtAddZip.Text = String.Empty
        txtAddLastName.Focus(Windows.UI.Xaml.FocusState.Keyboard)
    End Sub

    Private Async Sub SaveRecord(sender As Object, e As RoutedEventArgs)
        Dim newAddress As AddressInfo = New AddressInfo()
        newAddress.lastName = txtAddLastName.Text
        newAddress.firstName = txtAddFirstName.Text
        newAddress.fullName = txtAddFirstName.Text + " " + txtAddLastName.Text
        newAddress.company = txtAddCompany.Text
        newAddress.address1 = txtAddAddress1.Text
        newAddress.address2 = txtAddAddress2.Text
        newAddress.city = txtAddCity.Text
        newAddress.state = txtAddState.Text
        newAddress.zip = txtAddZip.Text
        addressBook.Add(newAddress)
        ' Choose Project > Manage Nuget Packages...   then install JSON
        ' Add: using Newtonsoft.Json;
        ' Add:  using Newtonsoft.Json.Serialization;
        Dim json = JsonConvert.SerializeObject(addressBook)
        Dim myStream As System.IO.Stream = Await _
                  myAppFolder.OpenStreamForWriteAsync(dataFileName, _
                  CreationCollisionOption.ReplaceExisting)
        Dim sw As StreamWriter = New StreamWriter(myStream)
        Await sw.WriteAsync(json)
        sw.Dispose()
        ClearAddScreen()
        LoadData()
    End Sub

    Private Sub CloseAdd(sender As Object, e As RoutedEventArgs)
        grdAdd.Visibility = Visibility.Collapsed
    End Sub

    Private Sub ShowAddNewEntry(sender As Object, e As RoutedEventArgs)
        grdAdd.Visibility = Windows.UI.Xaml.Visibility.Visible
    End Sub
End Class

Public Class AddressInfo
    Public Property lastName As String
    Public Property firstName As String
    Public Property fullName As String 'calculated field for data bound display purposes
    Public Property company As String
```

```
    Public Property address1 As String
    Public Property address2 As String
    Public Property city As String
    Public Property state As String
    Public Property zip As String
End Class
```