```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.IO;
 4  using System.Linq;
 5  using System.Runtime.InteropServices.WindowsRuntime;
 6  using Windows.Foundation;
 7  using Windows.Foundation.Collections;
 8  using Windows.UI.Xaml;
 9  using Windows.UI.Xaml.Controls;
10  using Windows.UI.Xaml.Controls.Primitives;
11  using Windows.UI.Xaml.Data;
12  using Windows.UI.Xaml.Input;
13  using Windows.UI.Xaml.Media;
14  using Windows.UI.Xaml.Navigation;
15  using muxc= Microsoft.UI.Xaml.Controls;
16  using System.Collections.ObjectModel;
17
18  // The Blank Page item template is documented at https://go.microsoft.com/
      fwlink/?LinkId=402352&clcid=0x409
19
20  namespace NavApp2
21  {
22      /// <summary>
23      /// ref: https://docs.microsoft.com/en-us/windows/uwp/design/controls-and-
        patterns/navigationview
24      /// Navigation view: Demo for the NavigationView control provides top-level
        navigation for your app.
25      /// It adapts to a variety of screen sizes and supports both top and left
        navigation styles.
26      /// This code shows how to use the Windows UI Library version of
        NavigationView, (Microsoft.UI.Xaml.Controls)
27      /// If you use the platform version of NavigationView instead,
        (Windows.UI.Xaml.Controls) the minimum version for your app project
28      /// must be SDK 17763 or greater. To use the platform version, remove all
        references to muxc:.
29      /// Windows UI Library = Microsoft.UI.Xaml.Controls
30      /// Platform version = Windows.UI.Xaml.Controls
31      /// Note! There are two versions of NavigationView, the Windows UI Library
        and the Platform version.
32      /// </summary>
33      public sealed partial class MainPage : Page
34      {
35
36          #region Fields/Properties
37
38          // List of ValueTuple holding the Navigation Tag and the relative
            Navigation Page
39          private readonly List<(string Tag, Type Page)> _pages = new List<(string
            Tag, Type Page)>
40          {
41              ("home", typeof(HomePage)),
42              ("apps", typeof(AppsPage)),
```

```
43              ("games", typeof(GamesPage)),
44              ("music", typeof(MusicPage)),
45              ("notes", typeof(NotesPage)),
46              ("mail", typeof(MailPage))
47
48          };
49
50          private double NavViewCompactModeThresholdWidth { get { return
              NavView.CompactModeThresholdWidth; } }
51
52          #endregion
53
54          #region Methods
55
56          /// <summary>
57          /// Class Constructor.
58          /// </summary>
59          public MainPage()
60          {
61              this.InitializeComponent();
62          }
63
64          /// <summary>
65          /// Called by the event handler for the NavigationView.BackRequested
              event.
66          /// </summary>
67          /// <returns></returns>
68          private bool On_BackRequested()
69          {
70              if (!ContentFrame.CanGoBack)
71                  return false;
72
73              //Don't go back if the nav pane is overlayed.
74              if (NavView.IsPaneOpen && (NavView.DisplayMode ==
                  muxc.NavigationViewDisplayMode.Compact ||
75                        NavView.DisplayMode ==
                           muxc.NavigationViewDisplayMode.Minimal))
76                  return false;
77
78              ContentFrame.GoBack();
79              return true;
80          }
81
82          /// <summary>
83          /// Navigate to selected Page.
84          /// </summary>
85          /// <param name="navItemTag"></param>
86          /// <param name="transitionInfo"></param>
87          private void NavView_Navigate(string navItemTag,
              Windows.UI.Xaml.Media.Animation.NavigationTransitionInfo
              transitionInfo)
88          {
```

```
89              Type _page = null;
90              if (navItemTag == "settings")
91              {
92                  _page = typeof(SettingsPage);
93              }
94              else
95              {
96                  var item = _pages.FirstOrDefault(p => p.Tag.Equals(navItemTag));
97                  _page = item.Page;
98              }
99              // Get the page type before navigation so you can prevent duplicate
100             // entries in the backstack.
101             var preNavPageType = ContentFrame.CurrentSourcePageType;
102
103             // Only navigate if the selected page isn't currently loaded.
104             if (!(_page is null) && !Type.Equals(preNavPageType, _page))
105             {
106                 ContentFrame.Navigate(_page, null, transitionInfo);
107             }
108         }
109
110         #endregion
111
112         #region Event Handlers
113
114         /// <summary>
115         /// NavigationView Loaded event
116         /// </summary>
117         /// <param name="sender"></param>
118         /// <param name="e"></param>
119         private void NavView_Loaded(object sender, RoutedEventArgs e)
120         {
121             // You can also add items in code.
122             NavView.MenuItems.Add(new muxc.NavigationViewItemSeparator());
123             NavView.MenuItems.Add(new muxc.NavigationViewItem
124             {
125                 Content = "My content",
126                 Icon = new SymbolIcon((Symbol)0xF1AD),
127                 Tag = "content"
128             });
129             _pages.Add(("content", typeof(MyContentPage)));
130
131             // Add handler for ContentFrame navigation.
132             ContentFrame.Navigated += On_Navigated;
133
134             // NavView doesn't load any page by default, so load home page.
135             NavView.SelectedItem = NavView.MenuItems[0];
136             // If navigation occurs on SelectionChanged, this isn't needed.
137             // Because we use ItemInvoked to navigate, we need to call Navigate
138             // here to load the home page.
139             NavView_Navigate("home", new                                    ↵
                  Windows.UI.Xaml.Media.Animation.EntranceNavigationTransitionInfo    ↵
```

```
                ());
140
141             // Add keyboard accelerators for backwards navigation.
142             var goBack = new KeyboardAccelerator { Key =
                  Windows.System.VirtualKey.GoBack };
143             goBack.Invoked += BackInvoked;
144             this.KeyboardAccelerators.Add(goBack);
145
146             // ALT routes here
147             var altLeft = new KeyboardAccelerator
148             {
149                 Key = Windows.System.VirtualKey.Left,
150                 Modifiers = Windows.System.VirtualKeyModifiers.Menu
151             };
152             altLeft.Invoked += BackInvoked;
153             this.KeyboardAccelerators.Add(altLeft);
154         }
155
156         /// <summary>
157         /// Handles the MainPage Keyboard "BackInvoked" event.
158         /// </summary>
159         /// <param name="sender"></param>
160         /// <param name="args"></param>
161         private void BackInvoked(KeyboardAccelerator sender,
              KeyboardAcceleratorInvokedEventArgs args)
162         {
163             On_BackRequested();
164             args.Handled = true;
165         }
166
167         /// <summary>
168         /// Handels the MainPage OnNavigatedTo event.
169         /// </summary>
170         /// <param name="sender"></param>
171         /// <param name="e"></param>
172         private void On_Navigated(object sender, NavigationEventArgs e)
173         {
174             NavView.IsBackEnabled = ContentFrame.CanGoBack;
175
176             if (ContentFrame.SourcePageType == typeof(SettingsPage))
177             {
178                 // SettingsItem is not part of NavView.MenuItems, and doesn't
                      have a Tag.
179                 NavView.SelectedItem = (muxc.NavigationViewItem)
                      NavView.SettingsItem;
180                 NavView.Header = "Settings";
181             }
182             else if (ContentFrame.SourcePageType != null)
183             {
184                 var item = _pages.FirstOrDefault(p => p.Page ==
                      e.SourcePageType);
185
```

```
186                 if ((item.Tag != "mail") && (item.Tag != "notes"))
187                 NavView.SelectedItem =
                      NavView.MenuItems.OfType<muxc.NavigationViewItem>().First(n =>
                      n.Tag.Equals(item.Tag));
188
189                 NavView.Header = ((muxc.NavigationViewItem)
                      NavView.SelectedItem)?.Content?.ToString();
190             }
191         }
192
193         /// <summary>
194         ///
195         /// </summary>
196         /// <param name="sender"></param>
197         /// <param name="args"></param>
198         private void NavView_ItemInvoked(muxc.NavigationView sender,
              muxc.NavigationViewItemInvokedEventArgs args)
199         {
200             if (args.IsSettingsInvoked == true)
201             {
202                 NavView_Navigate("settings",
                      args.RecommendedNavigationTransitionInfo);
203             }
204             else if (args.InvokedItemContainer != null)
205             {
206                 var navItemTag = args.InvokedItemContainer.Tag.ToString();
207                 NavView_Navigate(navItemTag,
                      args.RecommendedNavigationTransitionInfo);
208             }
209         }
210
211         /// <summary>
212         /// Handles the NavigationView BackRequested event
213         /// </summary>
214         /// <param name="sender"></param>
215         /// <param name="args"></param>
216         private void NavView_BackRequested(muxc.NavigationView sender,
              muxc.NavigationViewBackRequestedEventArgs args)
217         {
218             On_BackRequested();
219         }
220
221         /// <summary>
222         /// Handles the MainPage Navigation failure event: NavigationFailed
223         /// </summary>
224         /// <param name="sender"></param>
225         /// <param name="e"></param>
226         private void ContentFrame_NavigationFailed(object sender,
              NavigationFailedEventArgs e)
227         {
228             throw new Exception("Failed to load Page " +
                      e.SourcePageType.FullName);
```

```
229         }
230
231         #endregion
232     }
233
234 }
235
```