Simpler Page Navigation Framework for WP7

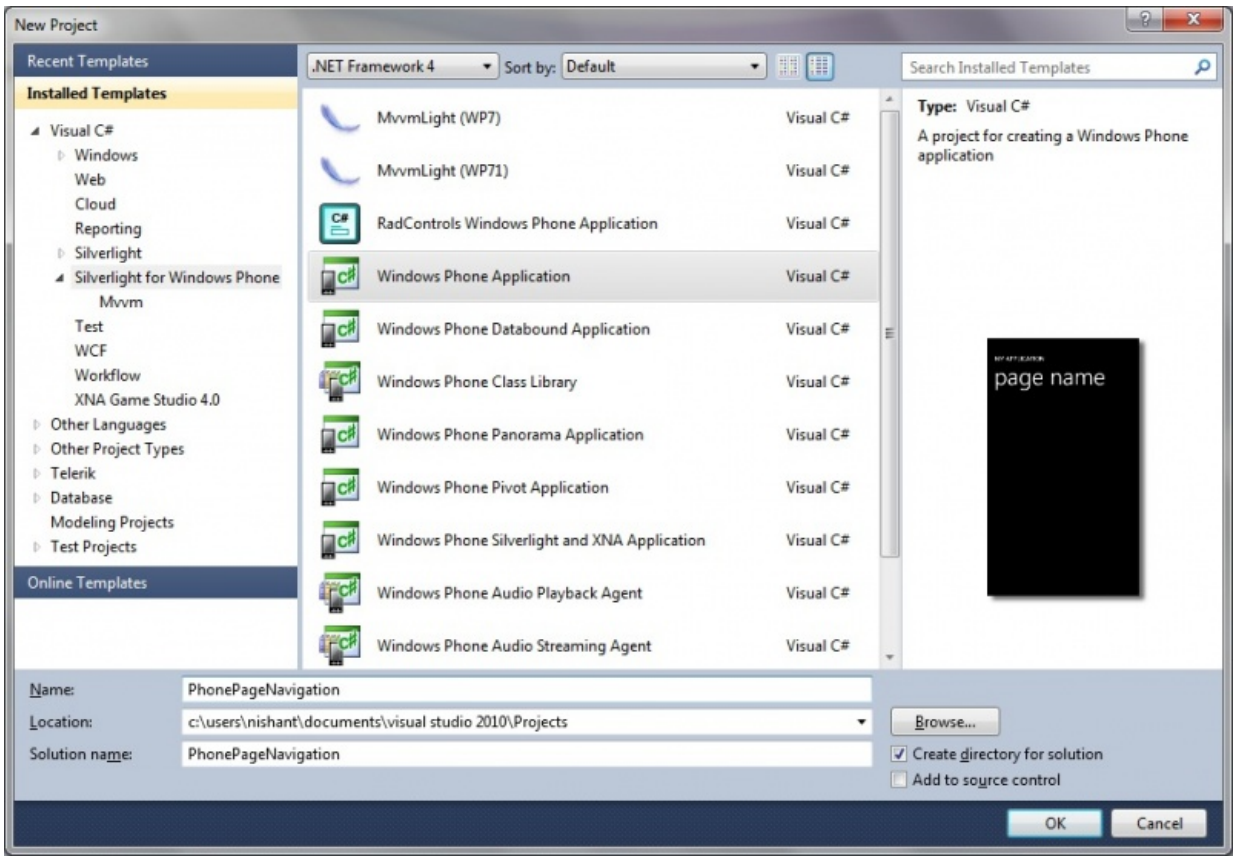
This article explains how to implement easy navigation between pages in Windows Phone 7 and Windows Phone 8 application.

Introduction

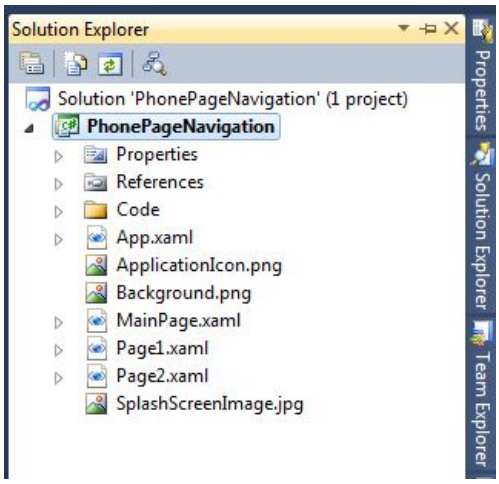
This is a sample application to demonstrate how to introduce easy navigation in your application. As your application grows, so does the number of pages in your application. But over a period of time if there is any change in the name of a page it might get very tedious to change the reference at all place. So, here is a simple solution for this. I'll walk through this methodology while creating a simple application.

Implementation

Step 1: Create a New project. Let's name this project "PhonePageNavigation"

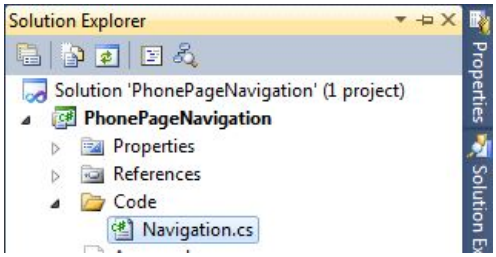


Step 2: Add 2 more pages Page1 and Page2 in your project. These pages will be used to demonstrate Navigation in WP7.



Step 3: Create a folder in your solution and name it "Code". Add a new class inside this folder call it "Navigation.cs".

(C) Copyright Nokia 2014. All rights reserved.



Step 4: Add an "enum" within the same file to declare each page distinctly. Your file structure would be like this (I am eliminating the using statements for saving space):

```
namespace PhonePageNavigation.Code
{
    public enum ApplicationPages
    {
        MainPage,
        Page1,
        Page2,
    }

    public static class Navigation
    {
    }
}
```

Step 5: Add an extension method to be used to navigate to a page. Usage of extension is useful since this method is by default available for all **Page** controls in you application.

```
public static void GoToPage(this PhoneApplicationPage phoneApplicationPage,
ApplicationPages applicationPage)
{
    switch (applicationPage)
    {
        case ApplicationPages.MainPage:
            phoneApplicationPage.NavigationService.Navigate(new Uri("/MainPage.xaml",
UriKind.Relative));
            break;
        case ApplicationPages.Page1:
            phoneApplicationPage.NavigationService.Navigate(new Uri("/Page1.xaml",
UriKind.Relative));
            break;
        case ApplicationPages.Page2:
            phoneApplicationPage.NavigationService.Navigate(new Uri("/Page2.xaml",
UriKind.Relative));
            break;
    }
}
```

Step 6: Add another overloaded extension method to pass param in your Navigation Uri (If in case you want to pass some parameters in your URI):

```
public static void GoToPage(this PhoneApplicationPage phoneApplicationPage,
ApplicationPages applicationPage, string parameter)
{
```


(C) Copyright Nokia 2014. All rights reserved.

```
    switch (applicationPage)
    {
        case ApplicationPages.MainPage:
            phoneApplicationPage.NavigationService.Navigate(new Uri("/MainPage.xaml?
key=" + parameter, UriKind.Relative));
            break;
        case ApplicationPages.Page1:
            phoneApplicationPage.NavigationService.Navigate(new Uri("/Page1.xaml?key=" +
parameter, UriKind.Relative));
            break;
        case ApplicationPages.Page2:
            phoneApplicationPage.NavigationService.Navigate(new Uri("/Page2.xaml?key=" +
parameter, UriKind.Relative));
            break;
    }
}
```

Step 7: Add 4 buttons to your main page to demonstrate working of this code.

Step 8: Add this code on MainPage.xaml.cs to handle some events.

```
private void btnGotoPg1_Click(object sender, System.Windows.RoutedEventArgs e)
{
    this.GoToPage(ApplicationPages.Page1);
}

private void btnGotoPg2_Click(object sender, System.Windows.RoutedEventArgs e)
{
    this.GoToPage(ApplicationPages.Page2);
```


(C) Copyright Nokia 2014. All rights reserved.

```
}

private void btnGotoPg1Param_Click(object sender, System.Windows.RoutedEventArgs e)
{
    this.GoToPage(ApplicationPages.Page1, "Text from Page 1");
}

private void btnGotoPg2Param_Click(object sender, System.Windows.RoutedEventArgs e)
{
    this.GoToPage(ApplicationPages.Page2, "Text from Page 1");
}
```

Step 9: Add this piece of code on page1.xaml.cs and page2.xaml.cs to handle some events.

```
protected override void OnNavigatedTo(System.Windows.Navigation.NavigationEventArgs e)
{
    base.OnNavigatedTo(e);
    try
    {
        IDictionary<string, string> parameters = this.NavigationContext.QueryString;
        if (parameters.ContainsKey("key"))
        {
            tbxSampleText.Text = parameters["key"];
        }
    }
    catch(Exception)
    {

    }
}

private void Button_Click(object sender, System.Windows.RoutedEventArgs e)
{
    if (NavigationService.CanGoBack)
        NavigationService.GoBack();
}
```

Working of the Sample

1. Now go to VS2010 and run the application in Emulator/Phone.

2. On your application MainPage press first button- "go to page 1" it will take you to the -"Page1" and the default text on your page will be displayed. Similarly "Page2" will be launched on pressing the second button - "go to page 2".

3. Pressing the button 3 and 4 i.e. "go to page 1 with uri" and "go to page 2 with uri" will launch "Page1" and "Page2" with text received from the main page and displayed on the screen.

4. Pressing the "go back" button on respective page will take you to the main page.

Explanation

Basically in step 5 and 6. We have created two Extension Methods'. Which will be available on each **Page congtrol**. These extension methods are accessible by using **this** keyword on each page. And we can Navigate to any page we want using these methods. We just need to update these methods each time a new page is added in our project.

A full working sample is attached in the following link.

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.