Windows® Phone

Pages and Navigation

Session 4.5

Topics

- Adding a new page to an application
- Page navigation
- Passing data between pages
- Using page navigation events
- Sharing objects between pages

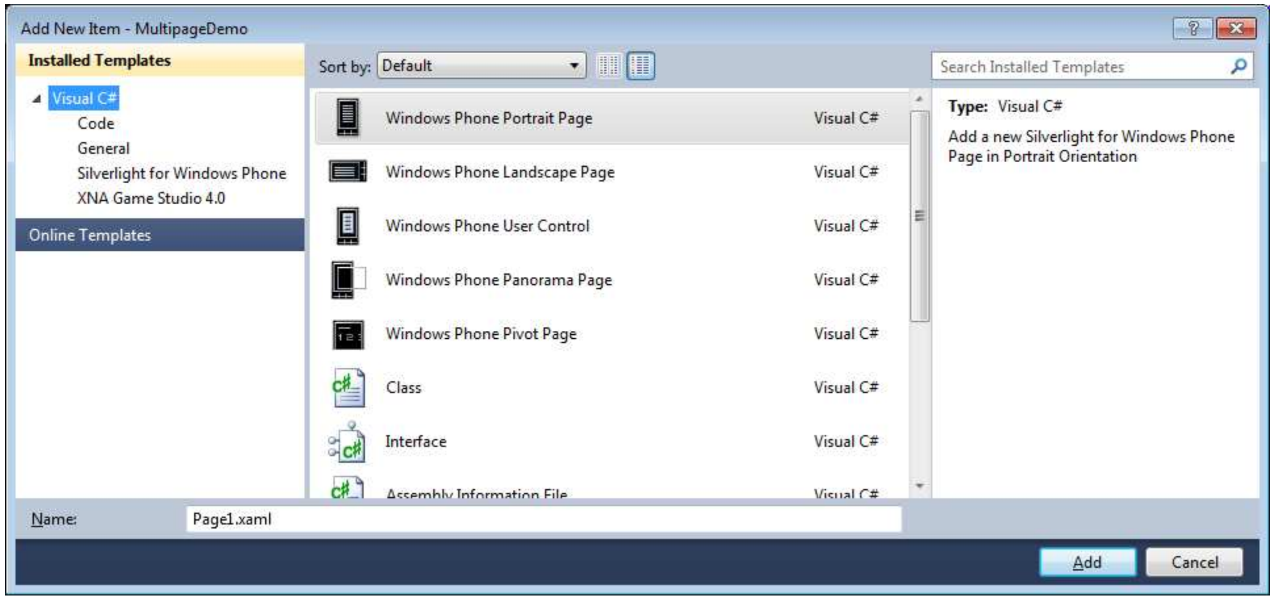
The Story So Far

- We are creating a simple Customer Manager application
- We have managed to display a list of customers in a ListBox and the customer can select a customer from this list
- Next we want to view the customer details on a separate page

Multi-page applications

- You often can't fit all the required elements of an application on a single page
  - Alternatively you might want to have separate "options" or "settings" pages in your application
- Silverlight on Windows Phone lets you add additional pages to an application and allow the user to navigate between them
- Before you write the code you should "storyboard" the application and forms

Adding another page


Add New Item - MultipageDemo

Installed Templates

- Visual C#
  - Code
  - General
  - Silverlight for Windows Phone
  - XNA Game Studio 4.0

Online Templates

Sort by: Default

Search Installed Templates

Windows Phone Portrait Page	Visual C#
Windows Phone Landscape Page	Visual C#
Windows Phone User Control	Visual C#
Windows Phone Panorama Page	Visual C#
Windows Phone Pivot Page	Visual C#
Class	Visual C#
Interface	Visual C#
Assembly Information File	Visual C#

Type: Visual C#

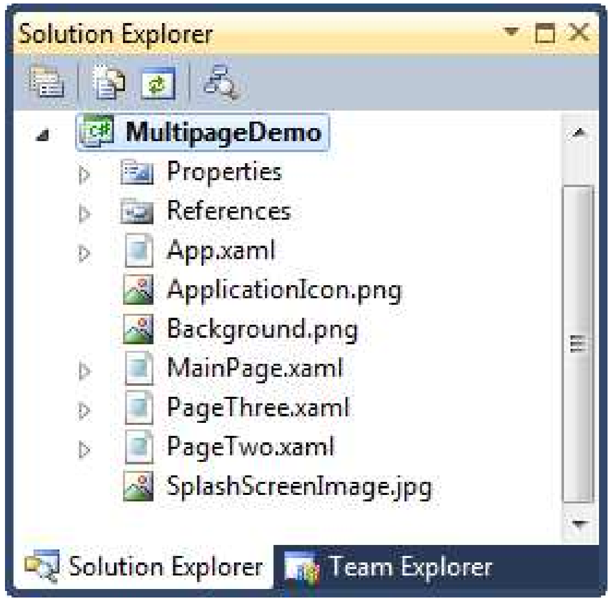
Add a new Silverlight for Windows Phone Page in Portrait Orientation

Name: Page1.xaml

- You add another page as you would any other item to a project
- This creates the xaml and the C# code file

Pages and projects

- Pages that are added are stored in the same project as the main page
- They will be compiled and transferred into the target device automatically


Solution Explorer

MultipageDemo
- Properties
- References
- App.xaml
- ApplicationIcon.png
- Background.png
- MainPage.xaml
- PageThree.xaml
- PageTwo.xaml
- SplashScreenImage.jpg

Page Navigation

```
private void page2Button_Click(object sender,
                               RoutedEventArgs e)
{
    NavigationService.Navigate(
                    new Uri("/CustomerDetailPage.xaml",
                        UriKind.RelativeOrAbsolute));
}
```

- The **`NavigationService`** object performs navigation for an application
- Each Silverlight page has a **`Uri`**
- The **`Navigate`** method takes the user to the specified page

The UriKind

```
private void page2Button_Click(object sender,
                               RoutedEventArgs e)
{
    NavigationService.Navigate(
                     new Uri("/CustomerDetailPage.xaml",
                         UriKind.RelativeOrAbsolute));
}
```

- The address of a resource can be expressed absolutely, or relative to the location the program is running from
- **`RelativeOrAbsolute`** will work, but the navigation in this case is actually **`Relative`**

Missing page exceptions

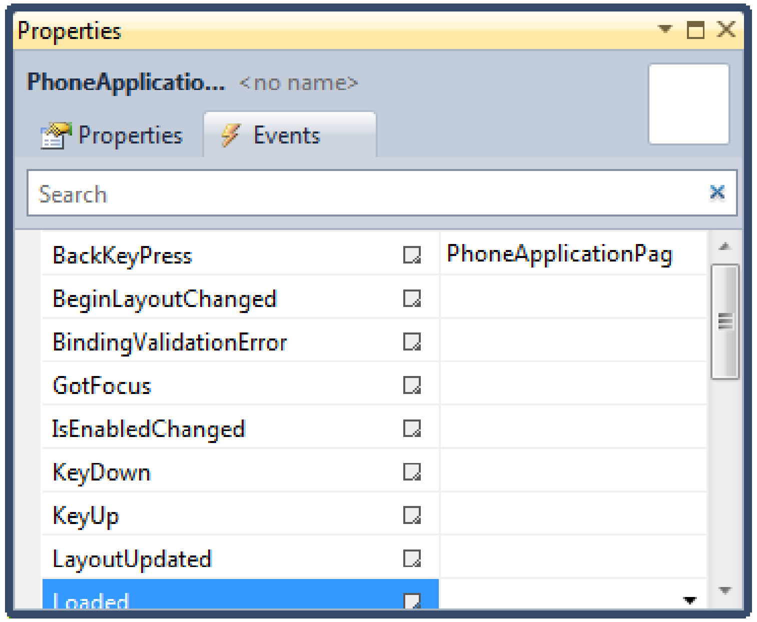
- The uri to the destination page is a string of text
- It is not a binding to a particular software object
- If you spell the uri incorrectly the Silverlight will compile correctly and start running
- The program will throw an exception if an incorrect uri is used to locate a page

Using the Back button

- The Back button is used on the Windows Phone to move backwards through the UI
- This behaviour is built into Silverlight page navigation on the device
- If the user presses Back they are automatically taken to the previous Silverlight page
- If they press Back at the top level of the pages the program is ended

Overriding the Back button

- You often want to get control when a user tries to move off a page
  - You might want to confirm a save at that point
- You can bind an event to the Back key pressed event which can do this



BackKeyPress	PhoneApplicationPag
BeginLayoutChanged	
BindingValidationError	
GotFocus	
IsEnabledChanged	
KeyDown	
KeyUp	
LayoutUpdated	
Loaded	

Disabling the Back Button

```
private void PhoneApplicationPage_BackKeyPress(
                  object sender,
                  System.ComponentModel.CancelEventArgs
e)
{
    e.Cancel = true;
}
```

- The event handler for the back button can cancel the back event
- Binding the above method to the back button stops it from allowing navigation away from a page

Using a MessageBox

```
private void PhoneApplicationPage_BackKeyPress(
                  object sender,
                  System.ComponentModel.CancelEventArgs e)
{
    if (MessageBox.Show("Do you really want to exit?",
                        "Page Exit",
                         MessageBoxButton.OKCancel)
       != MessageBoxResult.OK)
    {
        e.Cancel = true;
    }
}
```

- This code adds a confirmation message

Passing data between pages

- Each Silverlight page is a separate entity
- The code behind the page can hold data members but these are local to that page
- You often want to pass data from one page into another
    - We might want to send customer data into the CustomerDetailsPage
- If the data is very simple you can just add it to the uri that is used to locate the page

Assembling a data uri

```
// Get the selected customer from the list
Customer selectedCustomer = customerList.SelectedItem
                                  as Customer;
// Build a navigation string containing the information
    NavigationService.Navigate(
          new Uri("/CustomerDetailPage.xaml?" +
          "name=" + selectedCustomer.Name + "&" +
          "address=" + selectedCustomer.Address,
          UriKind.Relative));
```

- This event handler adds a data item onto the uri that is passed into the destination page
- There are two data elements, the name and the address string

Page navigated events

- If the destination page is to use the data in the uri there must be a way of running some code when a page is navigated to
- Silverlight provides methods to override in a page that give control when the page is navigated to and from
- We are going to use the **`OnNavigatedTo`** event

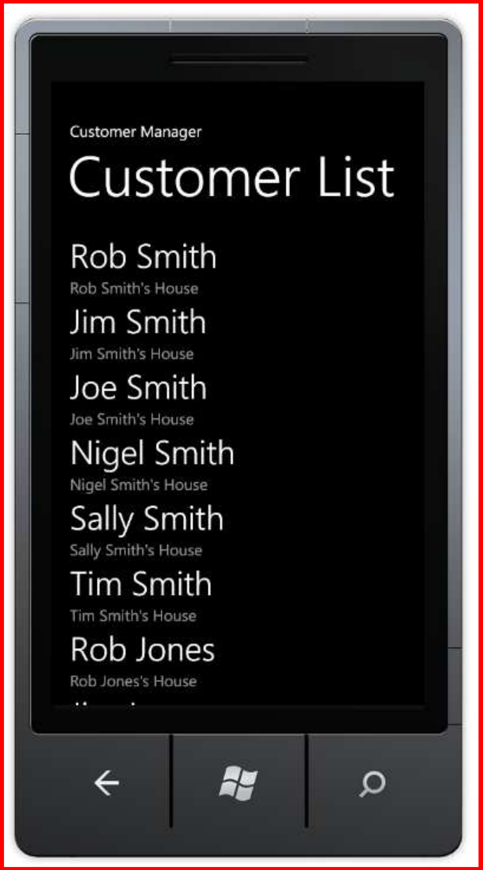
Loading data from the uri

```
protected override void OnNavigatedTo
         (System.Windows.Navigation.NavigationEventArgs e)
{
  string name, address;
  if (NavigationContext.QueryString.TryGetValue("name",
                                              out name))
      nameTextBlock.Text = name;
}
```

- The **`NavigationContext`** object has a **`QueryString`** property
- **`TryGetValue`** will search for a value in the **`uri`** and return true if it has found the value

Demo

Demo 2: Passing Data


Customer Manager

Customer List

Rob Smith

Rob Smith's House

Jim Smith

Jim Smith's House

Joe Smith

Joe Smith's House

Nigel Smith

Nigel Smith's House

Sally Smith

Sally Smith's House

Tim Smith

Tim Smith's House

Rob Jones

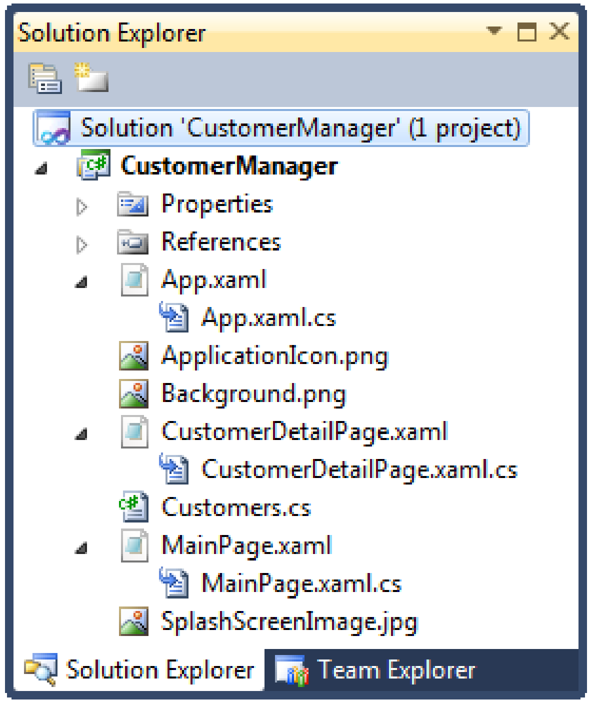
Rob Jones's House

Sharing objects between pages

- Each page in a program is a separate entity
- It has its own member data, but is not able to refer to any other
- When the program navigates to a page it does not provide a reference to the source page
- To share data amongst pages we have to use the **`App.xaml`** object

The App.xaml page

- This is the 'container' for the whole application
- It does not describe a page to be drawn, but it does hold methods that start the application running
- It also contains some event handlers



The App class

```
public partial class App : Application
{
    // To be used from all pages in the application
    public Customer ActiveCustomer;
}
```

- The **App** class for an application is an extension of the **`Application`** class provided by Silverlight
- We can add our own members to this class
- Here I have added an **`ActiveCustomer`** data member to track the active customer

Getting a reference to the App

```
protected override void OnNavigatedTo(
          System.Windows.Navigation.NavigationEventArgs e)
{
 base.OnNavigatedTo(e);

 // Get the parent App containing the active customer
 App thisApp = Application.Current as App;

 // Set the data context for the Grid to the selected
 // customer
 customerDisplayGrid.DataContext = thisApp.ActiveCustomer;
}
```

- The **Current** property of the **Application** class provides a reference to the currently active application

Getting a reference to the App

```
protected override void OnNavigatedTo(
          System.Windows.Navigation.NavigationEventArgs e)
{
 base.OnNavigatedTo(e);

 // Get the parent App containing the active customer
 App thisApp = Application.Current as App;

 // Set the data context for the Grid to the selected
 // customer
 customerDisplayGrid.DataContext = thisApp.ActiveCustomer;
}
```

- It is provided as a reference to the parent **Appplication** class

Getting a reference to the App

```
protected override void OnNavigatedTo(
          System.Windows.Navigation.NavigationEventArgs e)
{
 base.OnNavigatedTo(e);

 // Get the parent App containing the active customer
 App thisApp = Application.Current as App;

 // Set the data context for the Grid to the selected
 // customer
 customerDisplayGrid.DataContext = thisApp.ActiveCustomer;
}
```

- We need to convert this into an **App** reference so that we can access members of **App**

Getting a reference to the App

```
protected override void OnNavigatedTo(
          System.Windows.Navigation.NavigationEventArgs e)
{
 base.OnNavigatedTo(e);

 // Get the parent App containing the active customer
 App thisApp = Application.Current as App;

 // Set the data context for the Grid to the selected
 // customer
 customerDisplayGrid.DataContext = thisApp.ActiveCustomer;
}
```

- We need to convert this into an **App** reference so that we can access members of **App**

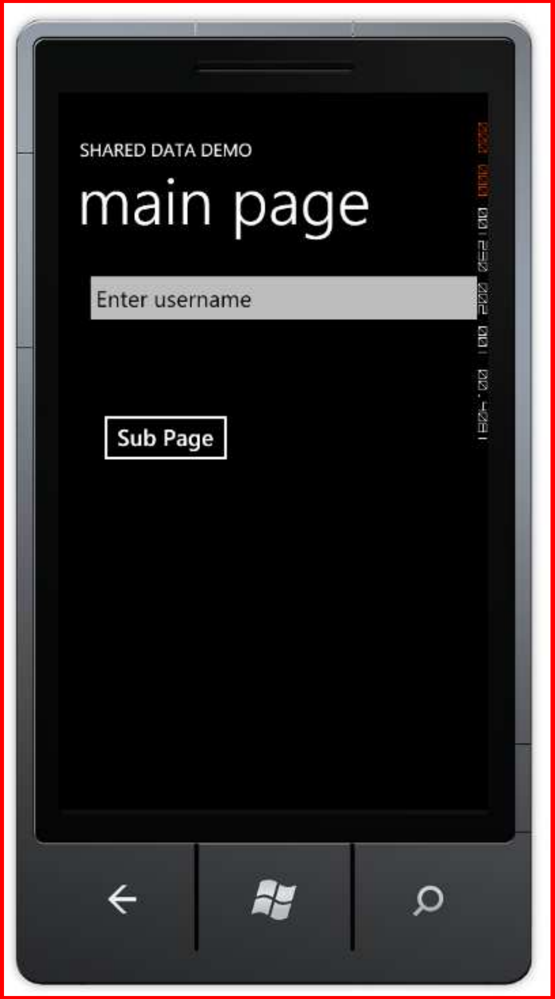
Setting the Edit Data Context

```
protected override void OnNavigatedTo(
          System.Windows.Navigation.NavigationEventArgs e)
{
 base.OnNavigatedTo(e);

 // Get the parent App containing the active customer
 App thisApp = Application.Current as App;

 // Set the data context for the Grid to the selected
 // customer
 customerDisplayGrid.DataContext = thisApp.ActiveCustomer;
}
```

- We then set the data context of the edit display to the currently active customer

Demo

Demo 3: Shared Data



Review

- You can create multiple Silverlight pages and add them to your project
- Navigation to pages is performed on the basis of uri (Uniform Resource Indicator) values
- The back button normally navigates back to the source page, but this can be overridden
- The uri can contain simple text messages
- Pages can share larger objects in the App.xaml page