ExpanderView in Silverlight for Windows Phone Toolkit

This code example explains how to use `ExpanderView` control from the Windows Phone Toolkit.

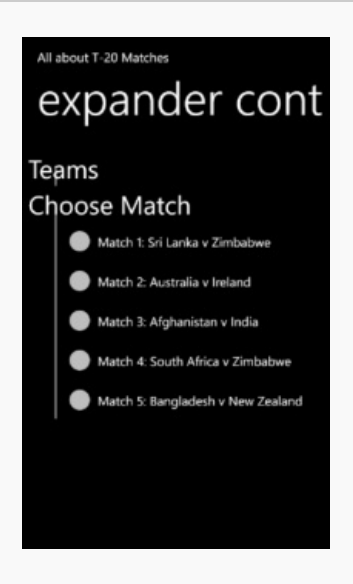
Introduction

`ExpanderView` comes with the Windows Phone Toolkit. It has a header which the user can toggle to expand/collapse additional content (this type of control is sometimes referred to as a collapsible panel or as an accordion control).

This article provides a brief overview of how to include and use the control in your projects. The example uses two `ExpanderView` components: the first displays a list of teams participating in a game, while the second allows you to select a specific game from a list.

Example app with expanders "closed"

"Teams" header expanded: view displays list of teams in cricket match



"Choose Match" expanded: view displays list of matches (checkable)

Example app with selected match

Adding the control to your project

First create a Windows Phone application:

- Open Visual Studio and select **Windows Phone Application** from the installed templates
- Select Windows Phone 7.1 as the Target Version.

- Right-click on the “References” in the project and click “Add Reference…”.Browse the “Microsoft.Phone.Controls.Toolkit.dll” and add it to the project.
- Add namespace of Microsoft.Phone.Controls.Toolkit in MainPage.xaml.

```
xmlns:toolkit="clr-
namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls.Toolkit"
```

Defining ExpanderView Control through XAML

The XAML code below shows how we create a `ExpanderView` named "Header 1", comprising of 8 `TextBlock` items.

```
 <toolkit:ExpanderView x:Name="Header1" Header="Teams " FontSize="40"
Expanded="Header1_Expanded"/>
                    <toolkit:ExpanderView.Items>
                        <TextBlock FontSize="20" Text="Afganistan"/>
                        <TextBlock FontSize="20" Text="Australia"/>
                        <TextBlock FontSize="20" Text="Bangladesh"/>
                        <TextBlock FontSize="20" Text="Ireland"/>
                        <TextBlock FontSize="20" Text="New Zealand"/>
                        <TextBlock FontSize="20" Text="South Africa"/>
                        <TextBlock FontSize="20" Text="Sri Lanka"/>
                        <TextBlock FontSize="20" Text="Zimbabwe"/>
                    </toolkit:ExpanderView.Items>
  </toolkit:ExpanderView>
```

Defining ExpanderView Control using CSharp

The same control can be created in C# as shown below:

- First we create an instance of `ExpanderView` (in this case called `Header1`):

```
ExpanderView Header1 = new ExpanderView();
```

- Then we set the header text, add the expander to a `stackpanel`, and set the content of the expander as a list.

```
public MainPage()
  {
  InitializeComponent();
  Header1.Header = "Expander Header";
  stackpanel.Children.Add(Header1);
  this.Header1.ItemsSource = new List<string>() { "Afganistan", "Australia",
"Bangladesh",
      "Ireland", "New Zealand", "South Africa", "Sri Lanka", "Zimbabwe" };
  }
```

Expanded and collapsed events

`ExpanderView` has two key events `Expanded` and `Collapsed` which are triggered when the state of `ExpanderView` is changed.

When the `Header1` `ExpanderView` is expanded its `Expanded` handler is called:

```
private void Header1_Expanded(object sender, RoutedEventArgs e)
  {
```

```
  // can define any specific action we want to take when the ExpanderView expands
  }
```

Similarly, when it is collapsed, the `Collapsed` handler is called:

```
private void Header1_Collapsed(object sender, RoutedEventArgs e)
  {
  // can define any specific action we want to take when the ExpanderView collapses
  }
```

Note: `ExpanderView` has `IsNonExpandable` property which makes the `ExpanderView` non expandable. This is useful if the expander can only be used in certain circumstances - perhaps after its content is fully populated.

Downloads

You can download sample project code from this file File:ExpanderControl.zip.