Custom splash screen with progress bar for Windows Phone applications

This article demonstrates how to create a splash screen with an animated progress bar on Windows Phone.

Introduction

The default splash screen in Windows Phone 7 application is a static image of the right size that you declare as a resource with name **splashscreenimage.jpg** and set as type "content" . This article shows how you can replace this splash screen with another - in this case one that contains an animated progress bar.

The new splashscreen uses the Popup class to display content on top of existing content. To remove "flicker" our new splashscreen will use the same static splashscreen as a background, but overlay it with the animated progress bar. The BackgroundWorker class is used to run startup operations in a separate thread, and when complete it closes the Popup class.

Splash Screen

Implementation

First create a project with *Windows Phone Application* Template. Once the project is being created, create a *Windows Phone User Control* **SplashScreenControl.xaml** and add `ProgressBar`, `TextBlock` and `Image` in it. The image is the default splash of the project.

```
<Grid x:Name="LayoutRoot" Background="White" Width="480" Height="800">
<ProgressBar HorizontalAlignment="Left" Margin="47,692,0,89" Name="progressBar1" Width="383"  />
<Image Height="512" HorizontalAlignment="Left" Margin="0,0,0,0" Name="image1" Stretch="Fill" VerticalAlignment="Top" Width="480" Sou
<TextBlock HorizontalAlignment="Left" Margin="185,656,0,114" Name="textBlock1" Text="Please Wait..." Width="111" Foreground="Black"
</Grid>
```

In *MainPage.xaml.cs* constructor we called a function `ShowSplash()` to load the popup.

```
private void ShowSplash()
        {
            this.popup = new Popup();
            this.popup.Child = new SplashScreenControl();
            this.popup.IsOpen = true;
            StartLoadingData();
        }
```

We initialize the Popup class and then set `SplashScreenControl` class to be hosted in the popup. `IsOpen()` opens the popup. Till now the code will load the splash screen (popup) with the progress bar.

 (C) Copyright Nokia 2013. All rights reserved.

Now we will add some background process, and when the background process gets completed we will close the popup as a result user will see the splash screen. Function `StartLoadingData()` starts and complete the background work.

```
private void StartLoadingData()
        {
            backroungWorker = new BackgroundWorker();
            backroungWorker.DoWork += new DoWorkEventHandler(backroungWorker_DoWork);
            backroungWorker.RunWorkerCompleted += new RunWorkerCompletedEventHandler(backroungWorker_RunWorkerCompleted);
            backroungWorker.RunWorkerAsync();
        }
```

First we initialize the BackgroundWorker class. We called `RunWorkerAsync()` function to starts execution of a background operation and thus `backroungWorker_DoWork()` is called.

```
void backroungWorker_DoWork(object sender, DoWorkEventArgs e)
        {
            //here we can load data
            Thread.Sleep(9000);
        }
```

Here we use the `Sleep()` function to wait for some time. When `backroungWorker_DoWork()` gets expired, `backroungWorker_RunWorkerCompleted()` is called which closes the popup.

```
void backroungWorker_RunWorkerCompleted(object sender, RunWorkerCompletedEventArgs e)
        {
            this.Dispatcher.BeginInvoke(() =>
            {
                this.popup.IsOpen = false;
            }
            );
        }
```

Thus user can see a splash screen with the progress bar.

To make a continuous progress bar we add the below code in *SplashScreenControl.xaml.cs*

```
this.progressBar1.IsIndeterminate = true;
```

Source Code

The full source code of the example is available here: File:SplashBar.zip

http://www.developer.nokia.com/Community/Wiki/Custom_splash_screen_with_progress_bar_for_Windows_Phone_applications (C) Copyright Nokia 2013. All rights reserved.

 (C) Copyright Nokia 2013. All rights reserved.