Collage using Nokia Imaging SDK

This article demonstrates how to create a "Collage" app with filter effects using Nokia Imaging SDK for Windows Phone 8.

Introduction

In this article we will add filter effects to our images using Nokia Imaging SDK and create a collage with collection of pictures.

Collage app

Implementation

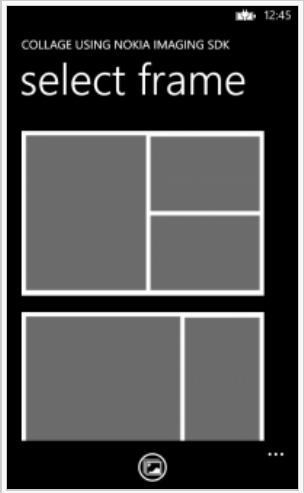
Create a standard Windows Phone Project

Let's create a standard Windows Phone Application Project

- Launch **Visual Studio**
- Click on **File**
- Click on **New Project**
- Select **Windows Phone App** (Visual C# Template)
- Add **Name** and **Location** of the project
- Click **OK** to create the project.

Adding UI controls on application page

- Let's add the below code in **MainPage.xaml** to display the number of frames that user can select from.



Collage frames

```
<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
    <Grid>
        <ListBox x:Name="ListBoxFrames" Margin="12,0,12,80" ItemsSource="{Binding}" Loaded="ListBoxCommon_Loaded_1" SelectionC
            <ListBox.ItemTemplate>
                <DataTemplate>
                    <StackPanel Orientation="Vertical" Margin="0,0,0,0">
                        <Image Stretch="Uniform" Source="{Binding Img_Frames}" Height="300" Width="400" HorizontalAlignment="C
                    </StackPanel>
                </DataTemplate>
            </ListBox.ItemTemplate>
        </ListBox>
    </Grid>
</Grid>
```


(C) Copyright Nokia 2013. All rights reserved.

- Create a menu bar in **MainPage.xaml** to display all created collages in one place.

```
<phone:PhoneApplicationPage.ApplicationBar>
        <shell:ApplicationBar IsVisible="True"  Opacity="0.1" IsMenuEnabled="False" Mode="Default" BackgroundColor="Black" For
            <shell:ApplicationBarIconButton IconUri="Assets/Icons/appbar.image.multiple.png" Text="my collage" Click="Applicat
        </shell:ApplicationBar>
</phone:PhoneApplicationPage.ApplicationBar>
```

- When the user selects a particular frame from the list it gives an option to edit the frame by tapping on it. It also allows user to add a suitable text on the collage. Please follow the **FrameEditPage.xaml** file from source code to create the the complete UI.

Frame Edit Page

```
<StackPanel Name="stack_Couple" Visibility="Collapsed" HorizontalAlignment="Left" VerticalAlignment="Top" Orientation="Horizon
    <StackPanel Name="stack_Couple_Image1" Height="480" Width="500" Tap="stack_Couple_Image1_Tap_1"  >
        <ScrollViewer Height="480" Width="500"  VerticalScrollBarVisibility ="Hidden" HorizontalScrollBarVisibility="Hidden">
            <ScrollViewer.Content>
                <StackPanel>
                    <slideView:PanAndZoomImage
                               ZoomMode="Free"
                               x:Name="Image1_Couple"
                               MaximumZoom="4"
                               Stretch="Fill"/>
                    <!--Image Name="Image1_Couple"   Stretch="Fill"  HorizontalAlignment="Center" VerticalAlignment="Center">
                    </Image-->
                </StackPanel>
            </ScrollViewer.Content>
        </ScrollViewer>
        <Border BorderBrush="White" BorderThickness="12" Height="480" Width="500" Margin="0,-480,0,0" >
        </Border>
    </StackPanel>
    <StackPanel Name="stack_Couple_Image2" Height="480" Width="312" Margin="-12,0,0,0"  Tap="stack_Couple_Image2_Tap_1">
        <ScrollViewer Height="480" Width="312"  VerticalScrollBarVisibility ="Hidden" HorizontalScrollBarVisibility="Hidden">
            <ScrollViewer.Content>
                <StackPanel>
                    <slideView:PanAndZoomImage
                               ZoomMode="Free"
                               x:Name="Image2_Couple"
                               MaximumZoom="4"
                               Stretch="Fill"/>
                    <!--Image Name="Image2_Couple"  Stretch="Fill"  HorizontalAlignment="Center" VerticalAlignment="Center">
                    </Image-->
                </StackPanel>
            </ScrollViewer.Content>
        </ScrollViewer>
        <Border BorderBrush="White" BorderThickness="10" Height="480" Width="312" Margin="0,-480,0,0" >
        </Border>
    </StackPanel>
</StackPanel>
```

Adding code behind

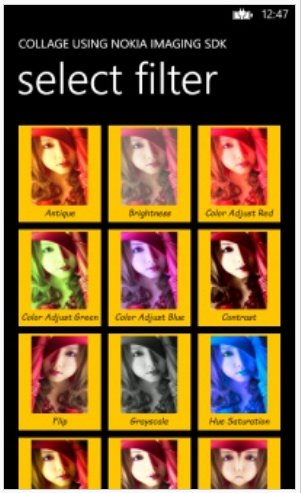
- We give user an option to put the original image onto the frames or add filters to the original image and then set it in the frames. We have also added an option to undo the filters if users wants to.

(C) Copyright Nokia 2013. All rights reserved.

Selected Picture

```
async void ImageEditorPage_Loaded(object sender, RoutedEventArgs e)
{
   ((ApplicationBarIconButton)(ApplicationBar.Buttons[2])).IsEnabled = false;
   if (App.PhotoModel != null)
   {
       await App.PhotoModel.RenderBitmapAsync(ImageProperties.iWriteableBitmap);
       if (App.PhotoModel.CanUndoFilter)
           ((ApplicationBarIconButton)(ApplicationBar.Buttons[2])).IsEnabled = true;
       else
           ((ApplicationBarIconButton)(ApplicationBar.Buttons[2])).IsEnabled = false;
   }
    selectedImage.Source = ImageProperties.iWriteableBitmap;
}
private void ApplicationBarIconButton_Click_1(object sender, EventArgs e)
{
    ImageProperties.bmpSelectedImage[FrameType.FrameSelectedIndex] = ImageProperties.iWriteableBitmap;
    NavigationService.GoBack();
}
private void ApplicationBarIconButton_Click_2(object sender, EventArgs e)
{
    NavigationService.Navigate(new Uri("/Filter/FilterPage.xaml", UriKind.RelativeOrAbsolute));
}
private async void ApplicationBarIconButton_Click_3(object sender, EventArgs e)
{
    App.PhotoModel.UndoFilter();
    if (App.PhotoModel != null)
    await App.PhotoModel.RenderBitmapAsync(ImageProperties.iWriteableBitmap);
    selectedImage.Source = ImageProperties.iWriteableBitmap;
    if (App.PhotoModel != null)
    {
        if (App.PhotoModel.CanUndoFilter)
            ((ApplicationBarIconButton)(ApplicationBar.Buttons[2])).IsEnabled = true;
        else
            ((ApplicationBarIconButton)(ApplicationBar.Buttons[2])).IsEnabled = false;
    }
}
```

- The **FilterPage.xaml** file displays the list of filters and when the user selects a particular filter it gets added to the selected image.



Filter Options

```
private async void FilterPage_Loaded(object sender, RoutedEventArgs e)
{
    await RenderAsync();
}
private async Task RenderAsync()
{
    if (!Busy)
    {
        Busy = true;
        int side = 136;
        try
        {
            await RenderThumbnailsAsync(ImageProperties.iWriteableBitmap, side, App.FilterModel.ArtisticFilters, StandardFilte
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.ToString());
            NavigationService.GoBack();
        }
        Busy = false;
    }
}
private async Task RenderThumbnailsAsync(WriteableBitmap bitmap, int side, List<FilterModel> list, WrapPanel panel)
{
    var fileStream = new MemoryStream();
    bitmap.SaveJpeg(fileStream, (int)bitmap.PixelWidth, (int)bitmap.PixelHeight, 100, 100);
    fileStream.Seek(0, SeekOrigin.Begin);
    App.PhotoModel = new PhotoModel() { Buffer = fileStream.GetWindowsRuntimeBuffer() };
    using (EditingSession session = await EditingSessionFactory.CreateEditingSessionAsync(fileStream))
```


(C) Copyright Nokia 2013. All rights reserved.

```
    {
        int i = 0;
        foreach (FilterModel filter in list)
        {
            WriteableBitmap writeableBitmap = new WriteableBitmap(side, side);
            foreach (IFilter f in filter.Components)
            {
                session.AddFilter(f);
            }
            Windows.Foundation.IAsyncAction action = session.RenderToBitmapAsync(writeableBitmap.AsBitmap());
            i++;
            if (i % 10 == 0)
            {
                // async, give control back to UI before proceeding.
                await action;
            }
            else
            {
                // synchroneous, we keep the CPU for ourselves.
                Task task = action.AsTask();
                task.Wait();
            }
            PhotoThumbnail photoThumbnail = new PhotoThumbnail()
            {
                Bitmap = writeableBitmap,
                Text = filter.Name,
                Width = side,
                Margin = new Thickness(6)
            };
            photoThumbnail.Tap += (object sender, System.Windows.Input.GestureEventArgs e) =>
            {
                App.PhotoModel.ApplyFilter(filter);
                App.PhotoModel.Dirty = true;
                NavigationService.GoBack();
            };
            panel.Children.Add(photoThumbnail);
            session.UndoAll();
        }
    }
}
```

- **MyCollageCollection.xaml** file displays all collages created by the user and also have the option to share on social network.

Collage collection

```
void MyCollageCollection_Loaded(object sender, RoutedEventArgs e)
{
    this.picturesList.ItemsSource = null;
    this.source.Clear();
    this.RefreshSavedImages();
    this.picturesList.ItemsSource = this.source;
}
private void RefreshSavedImages()
{
    using (MediaLibrary defaultLib = new MediaLibrary())
    {
        PictureAlbum savedPicturesAlbum = defaultLib.RootPictureAlbum.Albums.FirstOrDefault<PictureAlbum>(album => album.Name.
        if (savedPicturesAlbum == null)
        {
            return;
        }
        IEnumerable<Picture> pictures = savedPicturesAlbum.Pictures.Where<Picture>(pic => pic.Name.Contains("CollageNokiaImage
        foreach (Picture p in pictures)
        {
            this.source.Add(new PictureModel(p));
        }
    }
}
private void ListBoxCommon_Loaded_1(object sender, RoutedEventArgs e)
{
    ListBox listBox = (sender as ListBox);
    listBox.SelectedIndex = -1;
```


(C) Copyright Nokia 2013. All rights reserved.

```
}
private void picturesList_SelectionChanged_1(object sender, SelectionChangedEventArgs e)
{
    if (e.AddedItems.Count > 0)
    {
      object selectedItem = (PictureModel)e.AddedItems[0];
      int SelectedFrameIndex = ((ListBox)sender).SelectedIndex;
      ((ListBox)sender).SelectedIndex = -1;
      FrameworkElement root = Application.Current.RootVisual as FrameworkElement;
      root.DataContext = selectedItem;
      Deployment.Current.Dispatcher.BeginInvoke(() =>
        {
            ImageProperties.CollageImageName = ((PictureModel)DataContext).PictureName;
            ImageProperties.CollageImageSource = ((PictureModel)DataContext).PictureSource;
            NavigationService.Navigate(new Uri("/MyCollage/ShareMyCollage.xaml", UriKind.RelativeOrAbsolute));
        });
    }
}


public class PictureModel
{
    private WeakReference<ImageSource> picture;
    private Picture libraryModel;
    public PictureModel(Picture libPic)
    {
        this.libraryModel = libPic;
    }
    public Stream GetStream()
    {
        return this.libraryModel.GetImage();
    }
    private string _pictureName;
    public string PictureName
    {
        get { return this.libraryModel.Name; }
        set { _pictureName = value; }
    }
    public ImageSource PictureSource
    {
        get
        {
            ImageSource s = null;
            if (this.picture != null && this.picture.TryGetTarget(out s))
            {
                return s;
            }
            else
            {
                BitmapImage bim = new BitmapImage();
                using (Stream str = libraryModel.GetImage())
                {
                    bim.SetSource(str);
                }
                s = bim;
                this.picture = new WeakReference<ImageSource>(bim);
            }
            return s;
        }
 }
```

Summary

This is a demo app that uses the feature of Nokia Imaging SDK. Most of the filter effects code used in this article are referred from Filter Explorer sample project.

References

- Nokia Imaging SDK
- Nokia Imaging SDK : API reference
- Filter Explorer Sample Project
- Filter Effects Sample Project
- Real-time Filter Demo

Source Code

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

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.