Name of Experiment: Camera Operation(Flash Setting)	Exp No: WP8
Background: Student should have basic knowledge of C#.	
Summary: This experiment is again based on the camera operation, in this experiment user can set the Flash of the camera as well as he/she can set for Auto Flash.	
Learning Objective: To learn the camera feature Flash (ON/OFF/AUTO) during the capturing process.	
Target Platforms: This application is tested on Windows Phone Emulator and Windows Phone(Lumia 800)**.**	

Procedure:

Step1. Repeat all steps [1-9] as in experiment WP5.

Step2. Drag and Drop a button on the content panel as shown in UI design. Set the content of Button as "FLASH".[Refer fig. no 1]

Step3. Now, initialise a string named currentFlashMode before the MainPage constructor.[Refer source code Section]

Step4. Add this line "`FlashButton.Content = "Fl:" + cam.FlashMode.ToString().`" Inside the `cam_Initialized()` in order to know the current status of the Flash.

Step5. Add an FlashButton_Click event handler in the MainPage.xaml in order to perform the action on Button click.

Step6. Define the body for FlashButton_Click.[Refer the Source code section]

Step7. Save all changes made to the experiment.

Step8. Press F5, in order to deploy the application on the Windows Phone Emulator.[Refer fig. no 2,3,4]

Source Code	**Comments**

MainPage.xaml
 	
`<!--LayoutRoot is the root grid where all page content is placed-->`
 	
`<Grid x:Name="LayoutRoot" Background="Transparent">`
`<Grid.ColumnDefinitions>`
`<ColumnDefinition Width="640" />`
`<ColumnDefinition Width="160" />`
`</Grid.ColumnDefinitions>`
 	
`<Canvas x:Name="viewfinderCanvas" Width="640" Height="480"`
`HorizontalAlignment="Left" >`	

```
                <!--Camera viewfinder -->
                <Canvas.Background>
                    <VideoBrush x:Name="viewfinderBrush" />
                </Canvas.Background>
            </Canvas>

            <!--Button StackPanel to the right of viewfinder>-->
            <StackPanel Grid.Column="1" >
                <Button x:Name="ShutterButton" Content="CAPTURE"
Click="ShutterButton_Click" FontSize="26" FontWeight="ExtraBold"
Height="75" />
            <Button x:Name="FlashButton" Content="FLASH"
Click="FlashButton_Click" FontSize="26" FontWeight="ExtraBold"
Height="75" />
        </StackPanel>

            <!--Used for debugging >-->
            <TextBlock Height="40" HorizontalAlignment="Left"
Margin="8,428,0,0" Name="txtDebug" VerticalAlignment="Top" Width="626"
FontSize="24" FontWeight="ExtraBold" />
        </Grid>
```

← FLASH Button

MainPage.xaml.cs

```
using System.Windows;
using Microsoft.Phone.Controls;
using Microsoft.Devices;
using Microsoft.Xna.Framework.Media;
using System;

namespace WP7
{
    public partial class MainPage : PhoneApplicationPage
    {
        PhotoCamera cam;
        MediaLibrary library = new MediaLibrary();
        int savedCounter = 0;
        private string currentFlashMode;

        // Constructor
        public MainPage()
        {
            InitializeComponent();
        }

        protected override void
OnNavigatedTo(System.Windows.Navigation.NavigationEventArgs e)
        {
            //base.OnNavigatedTo(e);
            if (PhotoCamera.IsCameraTypeSupported(CameraType.Primary)
== true)
            {
                cam = new PhotoCamera();
                viewfinderBrush.SetSource(cam);
                cam.Initialized += new
System.EventHandler<CameraOperationCompletedEventArgs>(cam_Initialized
);
                cam.CaptureCompleted += new
System.EventHandler<CameraOperationCompletedEventArgs>(cam_CaptureComp
leted);
                cam.CaptureImageAvailable += new
```

```
System.EventHandler<ContentReadyEventArgs>(cam_CaptureImageAvailable);
            }
            else
            {
                Deployment.Current.Dispatcher.BeginInvoke(delegate()
                {
                    txtDebug.Text = "Camera is not available";
                });
               // ResButton.IsEnabled = false;
                ShutterButton.IsEnabled = false;
                FlashButton.IsEnabled = false;
            }
        }

        protected void OnNavigatedFrom(object sender,
NavigatingEventArgs e)
        {
            if (cam != null)
            {
                cam.Dispose();
                cam.Initialized -= cam_Initialized;
                cam.CaptureCompleted -= cam_CaptureCompleted;
                cam.CaptureImageAvailable -=
cam_CaptureImageAvailable;
            }
        }
        public void cam_Initialized(object sender,
Microsoft.Devices.CameraOperationCompletedEventArgs e)
        {
            try{
                if (e.Succeeded)
                {

Deployment.Current.Dispatcher.BeginInvoke(delegate()
                    {
                        txtDebug.Text = "Camera is Initialised";
                        FlashButton.Content = "Fl:" +
cam.FlashMode.ToString();

                    });
                }
            }
            catch(Exception ex){
                txtDebug.Text=ex.Message;
            }

        }
        public void cam_CaptureCompleted(object sender,
CameraOperationCompletedEventArgs e)
        {
            savedCounter++;
            Deployment.Current.Dispatcher.BeginInvoke(delegate()
            {
                txtDebug.Text = "Picture is saved for you";
            });
        }
        public void cam_CaptureImageAvailable(object sender,
Microsoft.Devices.ContentReadyEventArgs e)
        {
            string filename = savedCounter + ".jpg";
            library.SavePictureToCameraRoll(filename, e.ImageStream);
        }
```

```
        private void ShutterButton_Click(object sender,
System.Windows.RoutedEventArgs e)
        {
            if (cam != null)
            {
                cam.CaptureImage();
                //savedCounter++;
            }
            else
            {
                Deployment.Current.Dispatcher.BeginInvoke(delegate()
                {
                    txtDebug.Text = "Camera device is not available";
                });
            }


        }
        private void FlashButton_Click(object sender, RoutedEventArgs
e)
        {
            switch (cam.FlashMode)
            {
                case FlashMode.Off:
                    if (cam.IsFlashModeSupported(FlashMode.On))
                    {
                        // Specify that flash should be used.
                        cam.FlashMode = FlashMode.On;
                        FlashButton.Content = "Fl:On";
                        currentFlashMode = "Flash mode: On";
                    }
                    break;
                case FlashMode.On:
                    if
(cam.IsFlashModeSupported(FlashMode.RedEyeReduction))
                    {
                        // Specify that the red-eye reduction flash
should be used.
                        cam.FlashMode = FlashMode.RedEyeReduction;
                        FlashButton.Content = "Fl:RER";
                        currentFlashMode = "Flash mode:
RedEyeReduction";
                    }
                    else if (cam.IsFlashModeSupported(FlashMode.Auto))
                    {
                        cam.FlashMode = FlashMode.Auto;
                        FlashButton.Content = "Fl:Auto";
                        currentFlashMode = "Flash mode: Auto";
                    }
                    else
                    {   cam.FlashMode = FlashMode.Off;
                        FlashButton.Content = "Fl:Off";
                        currentFlashMode = "Flash mode: Off";
                    }
                    break;
                case FlashMode.RedEyeReduction:
                    if (cam.IsFlashModeSupported(FlashMode.Auto))
                    {
                        cam.FlashMode = FlashMode.Auto;
                        FlashButton.Content = "Fl:Auto";
                        currentFlashMode = "Flash mode: Auto";
```

← Flash button handler

← Flash:OFF

←FLASH:ON

```
                    }
                    else
                    {    am.FlashMode = FlashMode.Off;
                        FlashButton.Content = "Fl:Off";
                        currentFlashMode = "Flash mode: Off";
                    }
                    break;
                case FlashMode.Auto:
                    if (cam.IsFlashModeSupported(FlashMode.Off))
                    {
                        cam.FlashMode = FlashMode.Off;
                        FlashButton.Content = "Fl:Off";
                        currentFlashMode = "Flash mode: Off";
                    }
                    break;
            }

            this.Dispatcher.BeginInvoke(delegate()
            {
                txtDebug.Text = currentFlashMode;
            });
        }
    }
}
```

← FLASH:AUTO

Screenshots

Fig. no 1 UI design for the Exp. No WP8

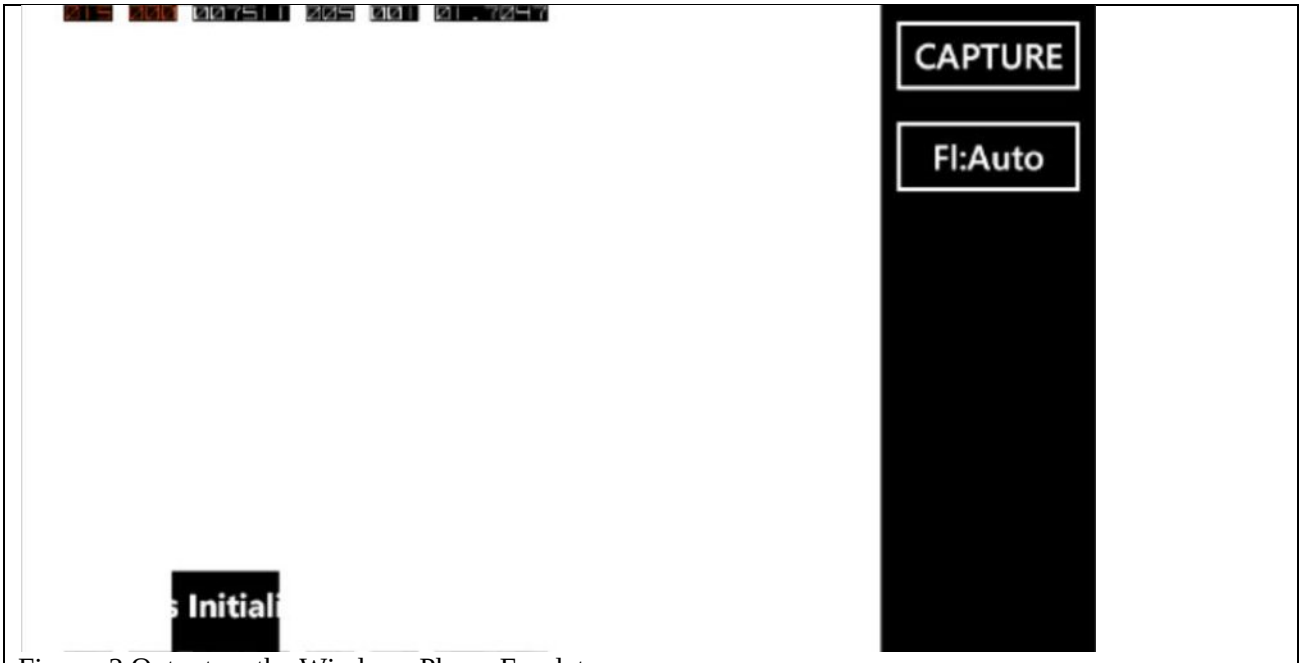

Fig. no 2 Output on the Windows Phone Emulator

Fig. no 3 Flash Button OFF

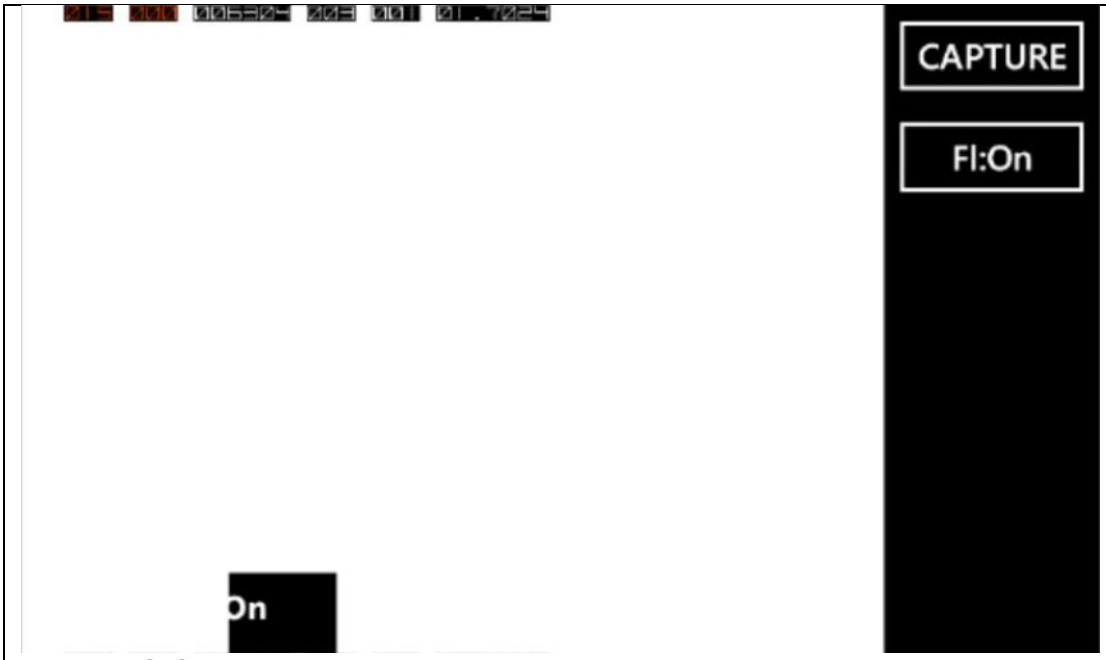

Fig. no 4 Flash Button ON

Observations:

It is observed that Flash Mode is very important in capturing the image, where the source of light is not enough for a good quality of Image.