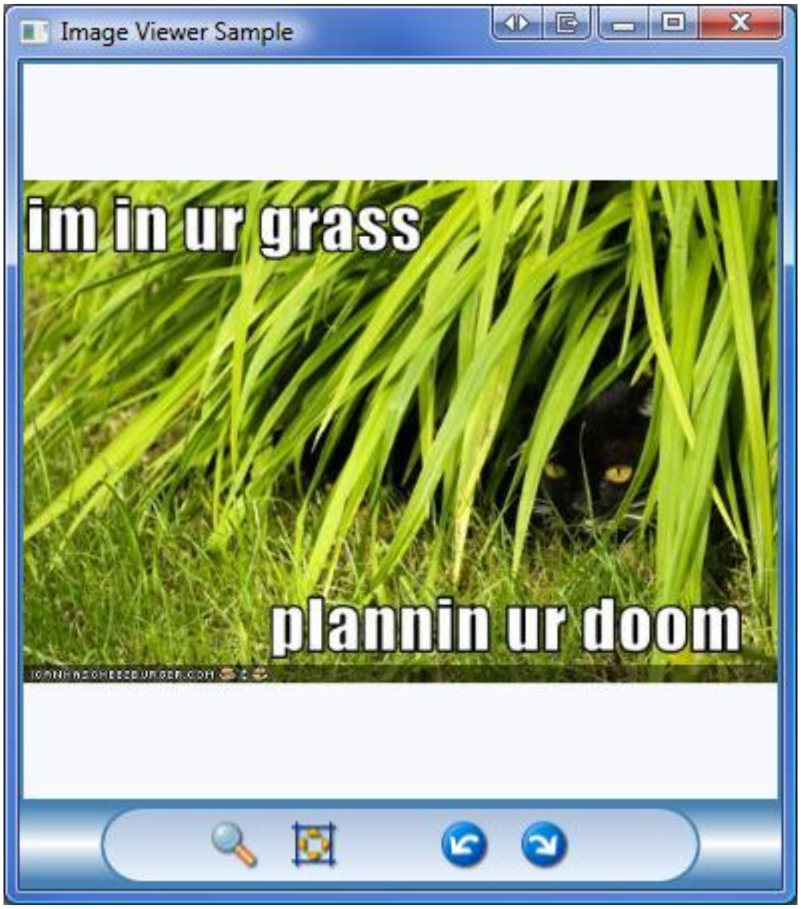
WPF: Simple Vista Image Viewer



[Download source code – 72.2 KB](#)
[Download sample app – 67.8 KB](#)

What does it do?

This lightweight WPF UserControl will display a single image in a manner similar to the Windows Photo Gallery. It doesn't manage more than one image in a gallery or slideshow fashion however, but it does:

- ***Zoom In/Out both with the mouse and with a slider control***
- ***Pan the image with the mouse***
- ***Rotate the image clockwise or counter-clockwise 90 degrees***
- ***Keep the image fully visible in the display area when not in zoom mode***

How can I use it?

You can use this image viewer in any WPF application with little effort since it's built as a WPF UserControl. By leveraging the ***WPF Interoperability*** *in WinForms it's possible to use this in a Windows Forms application as well, see the article here for instructions on accomplishing that.*
To get started using the control, download the source project from the above link. You can play around with the sample application to see an example of how to use the image viewer. The sample program simply provides a Window for the control and sets the path for the image to display; the control does all the rest.

To use the control in your own application you simply need to include the necessary source files in your own project or solution (CgsImageViewer.xaml(.cs), PanAndZoom.cs, and the images for the buttons). Once you've got them, you'll begin by inserting the viewer into your XAML, like so:

```
<Window x:Class="ImageViewer.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:iv="clr-namespace:ImageViewer"
    Title="Image Viewer Sample" Height="418" Width="410">
    <Grid>
        <iv:CgsImageViewer x:Name="cgsImageViewer" />
    </Grid>
</Window>
```

This is just placing the image viewer in a window and giving it a name. Now to load up an image all that you need to do is provide a path to the image or a BitmapImage object. You can do this in the business code or in the XAML by means of a data binding. Here's an example of setting the image in the business code:

```
cgsImageViewer.CurrentImagePath = @"c:\test images\cat-plans-your-doom.jpg";
```

That's all you really need to know to use the image viewer. Continue on for some extra info.

Class Members you might want to know about:

- ***CurrentImage*** *– Gets or sets the current BitmapImage used by the viewer.*
- ***CurrentImagePath*** *– Gets or sets a string representing the Uri to load an image from.*
- ***ZoomLevel*** *– Gets or sets a double representing the percentage to zoom.*
- ***ScaleToFit*** *– Resizes the image to fit within the visible bounds.*
- ***RotateLeft*** *– Rotates the image 90 degrees to the left and scales it to fit.*
- ***RotateRight*** *– Rotates the image 90 degrees to the right and scales it to fit.*

(Note: The code for panning and zooming was found here)

A little background and explanation:

I wrote this control in a grand total of about two days, which included testing. It was written for use in a WPF project I'm currently working on where users needed to be able to preview graphic files inside the program. I decided to make something that could be re-used and would provide simple viewing capabilities.

So here you go. If you’ve used it and like it why not leave a comment or trackback. If you find any bugs or have some ideas to improve on it let me know. I’m always looking to make things better!

- *Previous Entry:* *WPF: CheckBox as GroupBox Header*
- *Next Entry:* *Call of Duty: Modern Warfare 2 Multiple Profiles*