Tutorial – Drawing Images in GDI

For this tutorial we will be loading an image and then rendering it to the screen. After that, you will have all the skills you need in order to create a variety of different tools to help you develop your games.

The advanced exercises for this tutorial will build on the advanced tutorials from the previous two GDI sessions. If you have not yet completed them, you may wish to go back and finish them off so that you can combine them all together for this session

As with before, information regarding GDI+ can be found at:

https://msdn.microsoft.com/en-us/library/aa983623(v=vs.71).aspx

Getting ready to Render Text

1. Ensure you have overridden the Paint function for the applications Form, and that you have got a reference to the Graphics object currently in use. If you aren't sure how to do this, look over the exercise from previou sessions.

```
private void Form1_Paint(object sender, PaintEventArgs e)
 {
     //Get the Graphics Object that we will use for drawing
     Graphics g = e.Graphics;
```

2. Create a new Bitmap object inside the Form1 class. Since we do not want to be reloading the image every frame, we need to make sure that this is a member variable, and only loaded when the Form is first created.

```
public partial class Form1 : Form
    {
        private Bitmap m_bmpSmiley;
```

3. Download the "smiley.png" file from portal, or find another image that you would like to render. You can either place the file in the "Debug" folder in your build directory, or go to **Project Properties->Debug->Working Directory** to change the working directory, and place the file there.

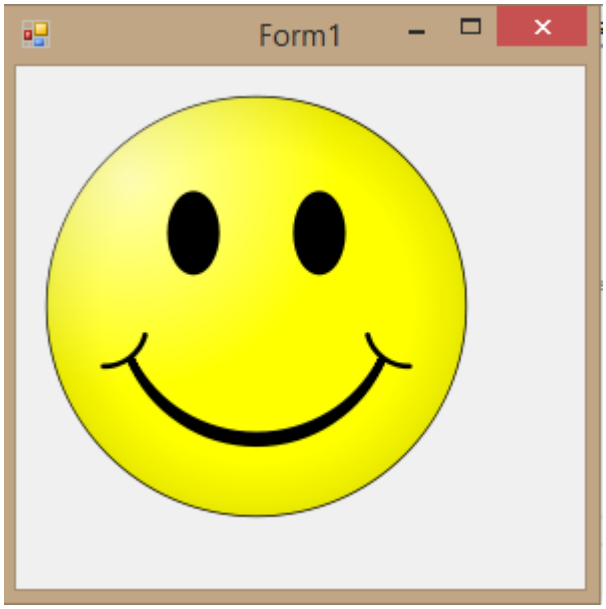
4. We next need to load the image in, by creating an instance of the Bitmap class, and passing in the filepath to the image being loaded.

```
public Form1()
{
    InitializeComponent();

    m_bmpSmiley = new Bitmap("./smiley.png");
}
```

5. Finally, we call DrawImage on the Graphics Object in order to actually render the image, using the Bitmap we previously loaded, and a Point which specifies where to render the image.

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    g.DrawImage(m_m_bmpSmiley, new Point(10, 10));
}
```

You should see something like the following when you run your application:



Some things to try:

1. Replace the cursor with a custom one. Use the **Cursor.Hide()** function to hide the mouse cursor when it enters the window, and instead draw an image at the cursor location instead.

   Don't forget to reshow the cursor if it goes out of the window bounds

2. Experiment with layering different bitmaps and experimenting with how drawing in different orders effects the final look of the Form.

Advanced:

Use what you have learned today to load in a sprite map from file, and display it to the user. Then, using what you have learned in previous sessions, add the ability for the user to click and drag to select different animation frames from within the sprite map, and label them with a frame number.

If you can get this working, you have begun working on the Sprite Tool required for your assessment.