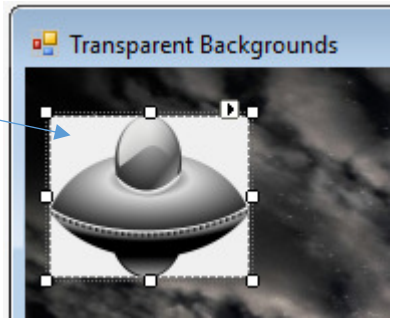
Transparent Backgrounds

There are multiple ways to have transparent portions of images in VB. If you intend on using Method 2, I recommend understanding how Timers, Key events and the coordinate system work.

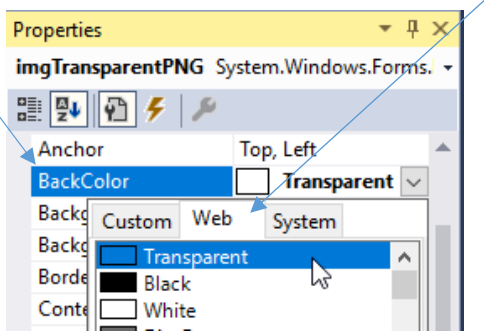
Method 1 – Using an Image File with a Transparent Background (easiest but with limitations)

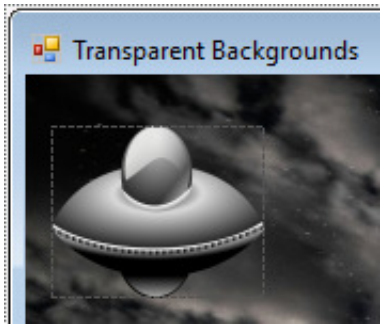
You can use photo-editing software to make parts of an image transparent, or just find and download an image with a transparent background.

Create a PictureBox, and add the image to it that has a transparent background. Notice that the transparent layers still do not appear transparent.

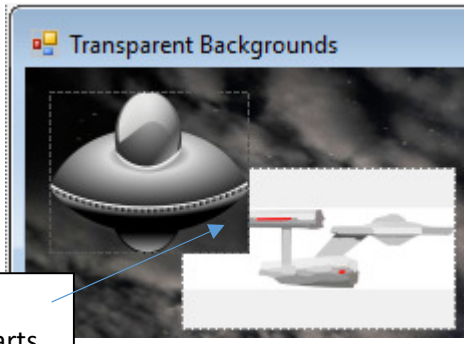


This is because you are seeing through the Image property and looking at the Background color of the PictureBox control. To fix this, you need to set the BackColor of your PictureBox to Transparent (in the **Web** tab).





Limitations: The problem with using this method is that what you see behind the transparent part of the PictureBox is whatever its Parent control is. By default this is the Form (unless you have put it in a Panel or another container). What this means is, if there is another PictureBox that you are overlapping, it will cover it up with the Background of the Form.



Notice that we are seeing the Form background through the transparent parts of the UFO PictureBox and not the PictureBox containing the Enterprise

Method 2 – Using a BitMap to Render a Graphic Directly on the Form (more difficult but you can overlap PictureBoxes)

This method adds more complexity because we cannot use PictureBoxes for your images. We are going to make BitMap objects to store the image and **paint** them on the Form using code. We will need to keep track of the Location and Size of the images in separate Rectangle objects. We also will not be able to see these images until we run our program.

Before we begin with the coding part, we need to edit the image file that we are going to use using image editing software.

Color the parts of the image file that you want to be transparent the same color, and make sure it is a color that doesn't appear anywhere else in the image that you don't want to be transparent. I recommend Magenta (255, 0, 255). I have provided a sample image of a UFO with this color. We will tell VB to make this specific color (or any color we wish) appear transparent in our program code.

At the top of your program, create a Bitmap object, and a Rectangle object for each image you wish to draw.

```
Dim ufo As Bitmap
Dim ufoBounds As New Rectangle(10, 10, 111, 81)
Dim meteor As Bitmap
Dim meteorBounds As New Rectangle(400, 100, 50, 100)
```

Width and **height** of image.

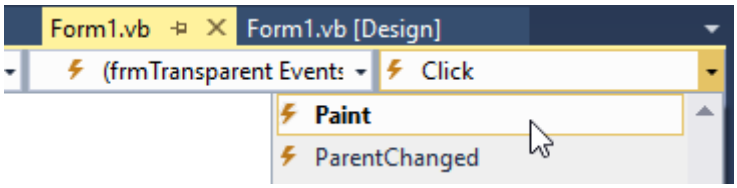
This defines where and how large our image will be.

(x, y) coordinate of top left of image.

We will need to add a Paint event to put the code that will actally draw these images.

In the Paint event, we will add the following code:

```
Dim g As Graphics = e.Graphics
g.DrawImage(ufo, ufoBounds)
g.DrawImage(meteor, meteorBounds)
```



This code will draw the Bitmaps onto the Form. This event will be called when the program first runs.

Animating with Bitmaps.

In the example program, I have provided some basic example code that uses a Timer, and Key events to move a ship and meteor so you can see how it can be done without PictureBoxes. It also has basic collision detection.

In order to call the Paint event to update the locations of your images, we need use the following line of code:

```
Me.Refresh()
```

This will re-paint the relevant parts of the Form. For smoother animation, set the **DoubleBuffered** property of the Form to **True**.