Learning 2D Game Programming: Images Part 3 - Blend Modes

Part 1, Part 2, Part 3

Table of Contents

Introduction

The last tutorial was about transforming the loaded image either via the SetScale or SetRotation function prior to drawing the image.

In this tutorial, we are going to learn a new concept call Blend Modes. A particular blend mode will have an impact on how an image will appear on the graphic screen.

Having said "image on graphic screen" we need to bear in mind that the drawing operations are normally to the **backbuffer** and only appears on the graphic screen when we issue the **flip** command.

SOLIDBLEND and LIGHTBLEND

The first blend mode we will look at is called SOLIDBLEND. It is the easiest to understand as when drawing in this mode all it does is simply writes over what ever is on the backbuffer.

To set the blending mode we use the SetBlend function as follows:-

```
Function SetBlend( blend )
```

Let us look at how this function works. Move the green rectangle over the red one and you see the green rectangle overwriting the red one.

```
Graphics 640,480
SetBlend  SOLIDBLEND
Repeat
  Cls
  SetColor 255,0,0
  DrawRect  100,100,100,100
  SetColor 0,255,0
  DrawRect MouseX(),MouseY(),100,100
  Flip
Until KeyHit(key_escape) Or AppTerminate()
End
```

Now change the blend mode to LIGHTBLEND which adds the pixels togther instead of overwriting them

```
SetBlend  LIGHTBLEND
```

Notice the two different effects. SOLIDBLEND simply overwrites whereas LIGHTBLEND adds the red pixels (which was already there) with the green pixel to give the yellow pixels (red+green=yellow) on the overlapped section.

Alpha Transparency, SetAlpha and ALPHABLEND

An image can be displayed on the screen with transparency effect using two functions in BlitzMax. One is the SetAlpha function which sets the transparency level (Zero being fully transparent and One being totally solid). In order to see the transparency effect, we also need to use the ALPHABLEND blend mode.

```
Graphics 640,480
SetBlend  ALPHABLEND
Local A:Float=1.0

Repeat
  Cls
  SetColor 255,0,0
  SetAlpha 1.0
  DrawText "Alpha="+a,10,10
  DrawRect  100,100,100,100
  SetColor 0,255,0
  SetAlpha A
  DrawRect  MouseX(),MouseY(),100,100

  If KeyDown(KEY_A)
    a :+ .25
    If a>1 Then a=0
  EndIf
  FlushKeys
  Flip

Until KeyHit(key_escape) Or AppTerminate()
End
```

By pressing the A key you can cycle through the five possible SetAlpha states of Zero, 0.25, .5, 0.75 and 1.0. See the images below for settings of 0.25 and 0.75 with different levels of transparency.

Transparency via Masking

Masking is like transparency except that you can only have two level, totally transparent or totally solid. The way masking works is that before a pixel is drawn to the backbuffer, it is first compared against a mask color. If it is the same as the mask color then it will not get drawn.

Let us demonstrate this effect via this program. The blendmode we are using this time is called MASKBLEND.

```
Graphics 640,480

Local image:TImage=CreateImage(100,100,1)
SetColor 255,0,0
DrawRect 0,0,100,100
SetColor 255,255,0
DrawOval 25,25,50,50
GrabImage Image,0,0

SetBlend MASKBLEND

Repeat
  Cls
  SetColor 0,255,255
  DrawRect 100,100,200,200
  SetColor 255,255,255
  DrawImage Image,MouseX(),MouseY()
  Flip
Until KeyHit(key_escape) Or AppTerminate()
End
```

Run the above program and then re-run it another time, this time setting the mask color to yellow:-

```
Graphics 640,480
SetMaskColor 255,255,0
Local image:TImage=CreateImage(100,100,1)
```

You will see that in the second case the yellow circle in the red rectangle is now totally transparent as the yellow color is now masked out.

Now change the MaskColor to red (SetMaskColor 255,0,0), think about what should happen and then run the program to see what happens.

Blend Modes and Images

So far what I have shown you are the various effects that blend modes have on your image when drawn onto the graphic screen.

Now let us see some practical application of these blend modes.

```
Graphics 640,480
Local Rocket:TImage=LoadImage(BlitzMaxPath()+"\samples\hitoro
\gfx\boing.png")
SetBlend MASKBLEND
Repeat
  Cls
  DrawRect 100,100,100,100
  DrawImage Rocket,MouseX(),MouseY()
  Flip
Until KeyHit(key_escape) Or AppTerminate()
End
```

As before, run the program with and without setmaskcolor and see what happens. The maskcolor this time is pink

```
Graphics 640,480
SetMaskColor 255,0,255
Local Rocket:TImage=LoadImage(BlitzMaxPath()+"\samples\hitoro
\gfx\boing.png")
```

Notice the difference between the two image. In the seond case, the pink portion of the image is now "transparent" or no longer displayed. This would be the sort of thing you want for your game instead of the image of spaceship in a pink rectangle.

Blend Modes and Images with Alpha Channel

In the previous section we saw how we can use a particular color of an image and turn that into a transparent color by using **SetMaskColor** and the **MASKBLEND** blending mode.

An image, as we saw earlier, derived its color from the values of its RGB attributes. Some image formats such as png, employs a fourth attribute, called the alpha channel, where the value of this attribute determines how a pixel should be displayed (alpha values are from zero to one).

Images with Alpha channels can be displayed using wither the MASKBLEND or ALPHABLEND mode.

When specifying a MASKBLEND blending mode, images whose alpha attributes are less than 0.5 are treated as transparent and apha values of greater than 0.5 are displayed solidly.

On the other hand, when specifying ALPHABLEND, the entire continuum of transparency between zero and one is available with the transparency decreasing from zero to 1 (zero totally transparent and one totally solid).

Execute the program below with ALPHABLEND and MASKBLEND to see the difference.

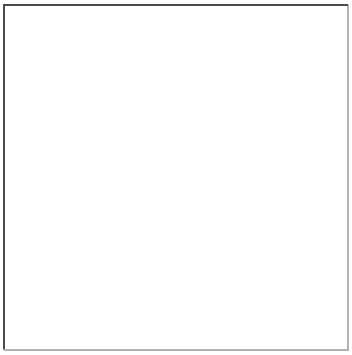
```
AutoMidHandle True
Local Player:TImage=LoadImage(LoadBank("http::www.w3.org/Graphics
/PNG/alphatest.png"))
SetBlend ALPHABLEND
SetClsColor 255,255,255

Repeat
  Cls
  SetColor 0,255,255
  DrawRect 50,50,100,100
  SetColor 255,0,0
  DrawRect 200,200,100,100
  SetColor 0,0,255
  DrawRect 400,300,100,100
  DrawImage Image,MouseX(),MouseY()
  Flip

Until KeyHit(key_escape) Or AppTerminate()
End
```

Notice how with MASKBLEND, the transparency is binary, whereas the ALPHABLEND is more graduated

Some images (or sprites) typically of png format are created with an alpha channel. Look at this image which I have converted the pink portion of it into a png image with transparency. This type of images can be readily loaded into BlitzMax and used with either ALPHABLEND or MASKBLEND modes.



So we do not need to specify a MaskColor when we use an image file with alpha channel..

```
Graphics 640,480
Local URL:String="http::www.2dgamecreators.com/tutorials/gameprogramming
/images/"
Local Rocket:TImage=LoadImage(LoadBank(URL+"alphaboing.png"))
SetBlend MASKBLEND
Repeat
  Cls
  DrawRect 100,100,100,100
  DrawImage Rocket,MouseX(),MouseY()
  Flip
Until KeyHit(key_escape) Or AppTerminate()
End
```

SetColor and SetClsColor

I thought I'll just mention these two commands. We have been using them a lot. SetClsColor as the name suggests set the color for the clearscreen function.

SetColor sets the color of the ink to draw with and has an impact on the color of the image to be drawn:-

```
Graphics 640,480
Local URL:String="http::www.2dgamecreators.com/tutorials/gameprogramming
/images/"
Local Rocket:TImage=LoadImage(LoadBank(URL+"alphaboing.png"))
SetBlend ALPHABLEND
Repeat
  Cls
  SetColor 255,255,255
  DrawImage Rocket,MouseX(),MouseY()
  SetColor 255,0,0
  DrawImage Rocket,MouseX()+150,MouseY()
  SetColor 0,0,255
  DrawImage Rocket,MouseX()+300,MouseY()

  Flip
Until KeyHit(key_escape) Or AppTerminate()
```

```
End
```

Note that **SetColor 255,255,255** draws an untinted image.

SetAlpha and GetAlpha

The Alpha setting (using **SetAlpha**) does not care whether an image has an alpha channel or not. If we take our earlier example, and change the alpha setting and use ALPHABLEND we get the following image:-

```
Graphics 640,480
Local Rocket:TImage=LoadImage(BlitzMaxPath()+"\samples\hitoro
\gfx\boing.png")
SetBlend ALPHABLEND
Repeat
  Cls
  SetAlpha 1.0
  DrawRect 100,100,100,100
  SetAlpha 0.45
  DrawImage Rocket,MouseX(),MouseY()
  Flip
Until KeyHit(key_escape) Or AppTerminate()
End
```

The first image with a SetAlpha of 1.0 (totally solid) and the second image with a SetAlpha of 0.35 (35% transparent)

Summary

In this tutorial we saw how the various blend modes can have an impact on the way the images are drawn to the graphic screen. Max2D recognises 5 blendmodes (SOLIDBLEND, MASKEDBLEND, ALPHABLEND, SOLIDBLEND and SHADEBLEND). These blendmodes can be mixed using the **|** (OR) operator.

We did not cover SHADEBLEND but pls feel free to experiment with it.

We also saw that Max2D can handle transparency not just with Masks but also with the new alpha transparency format.

There are other image topics to be covered but that will be for another day. In the meantime, happy programming :)

Back to the index, Previous Tutorial