Learning 2D Game Programming: Images Part 2 - Transformations

Part 1, Part 2, Part 3

Table of Contents

Introduction

In the previous tutorial we saw how images can be loaded into BlitzMax and displayed on a graphic screen. In this tutorial, we are going to see how we can further manipulate the loaded images and display these transformed images.

SetScale, GetScale

The first function we are going to look at is the SetScale function which has the following syntax:-

```
Function SetScale( scale_x#,scale_y# )
```

Let us look at how this function works. Press the X and Y keys to change the respective scaling factors.

```
Graphics 640,480
Local URL:String="http::www.2dgamecreators.com/tutorials
/gameprogramming/basic/"
Local Player:TImage=LoadImage(LoadBank(URL+"blobship_1-1.png"))
Local  x:float=1.0
Local  y:float=1.0

Repeat
  Cls
  SetScale 1.0,1.0
  DrawText "Scale X="+x+" Scale Y="+y,10,10

  SetScale x,y
  DrawImage Player,MouseX(),MouseY()

  If KeyDown(KEY_X)
     x :+ .5
     If x> 1.5 Then x=0.5
  EndIf
  If KeyDown(KEY_Y)
     y :+ .5
     If y> 1.5 Then y=0.5
  EndIf
  Flushkeys
  Flip
Until KeyHit(key_escape) Or AppTerminate()
End
```

At a scale factor of 1, the images is drawn at its original size, the other images will be at their respective scaling factors.

Now run the program below. Notice that negative values will cause the image to flip on the respective axis

```
Graphics 640,480
Local URL:String="http::www.2dgamecreators.com/tutorials
/gameprogramming/basic/"
Local Player:TImage=LoadImage(LoadBank(URL+"blobship_1-1.png"))
Local  x:float=1.0
Local  y:float=1.0

Repeat
  Cls
  SetScale 1.0,1.0
  DrawText "Scale X="+x+" Scale Y="+y,10,10

  SetScale x,y
  DrawImage Player,MouseX(),MouseY()

  If KeyDown(KEY_X)
     x =-x
  EndIf
  If KeyDown(KEY_Y)
     y =-y
  EndIf

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

If you need to know the current scale factor, use the GetScale function

```
Function GetScale( scale_x# Var,scale_y# Var )
```

SetRotation, GetRotation

The other transformation function is **SetRotation** and as the name suggests, rotates the image to be displayed:-

```
Function SetRotation( rotation# )
```

Just a simple example to demonstrate how this works

```
Graphics 640,480
AutoMidHandle True
Local URL:String="http::www.2dgamecreators.com/tutorials
/gameprogramming/basic/"
Local Player:TImage=LoadImage(LoadBank(URL+"blobship_1-1.png"))
Local  Rotation:float=0.0

Repeat
  Cls
  SetRotation 0.0
  DrawText "Current Rotation is "+Rotation

  SetRotation Rotation
```

```
      DrawImage Player,MouseX(),MouseY()

      If KeyDown(KEY_X)
         Rotation :+ 30
         If Rotation>360 Then Rotation=0.0
      EndIf
      Flushkeys
      Flip
    Until KeyHit(key_escape) Or AppTerminate()
    End
```

The thing to note here is that the handle of the image plays a big role in how the rotation is calculated. Note that for this example I have set the handle to be in the middle via the **AutoMidHandle True** statement.

Just like **GetScale**, there is also an equivalent **GetRotation** function.

SetTransform

This function is nothing more than to allow users to set both scale and rotation at the same time:-

```
Function SetTransform( rotation#=0,scale_x#=1,scale_y#=1 )
```

DrawImageRect

The last transformation function we are going to look at is the DrawImageRect. This function causes a lot of confusion as in previous Blitz incarnations, it draws to a rectangular image, almost like a setviewport function. In BlitzMax what this function does is that it takes a loaded image and then scales it to the rectangular area that you specify:-

```
Function DrawImageRect( image:TImage,x#,y#,w#,h#,frame=0 )
```

```
    Graphics 640,480
    AutoMidHandle True
    Local URL:String="http::www.2dgamecreators.com/tutorials
    /gameprogramming/basic/"
    Local Player:TImage=LoadImage(LoadBank(URL+"blobship_1-1.png"))
    Local X:Int=32
    Local y:Int=32

    Repeat
      Cls
      DrawText "X="+x+" y="+y,10,10
      DrawImageRect Player,MouseX(),MouseY(),x,y

      If KeyDown(KEY_X)
         x:+32
         If x>128 Then x=32
      EndIf
      If KeyDown(KEY_Y)
         y:+32
         If y>128 Then y=32
      EndIf

      FlushKeys
      Flip
    Until KeyHit(Key_Escape) Or AppTerminate()
    End
```

LoadImage: FILTEREDIMAGE and MIPMAPPEDIMAGE

In the last tutorial we ignored the flag parameter of the LoadImage function. To recap its syntax:-

```
Function LoadImage:TImage( url:Object,flags=-1 )
```

As we have not specified anything, the default value of -1 was used. We are going to look at 2 possible flags we can use:-

Flags value	Effect
FILTEREDIMAGE	The image is smoothed when scaled up to greater than its original size, when rotated, or when drawn at fractional pixel coordinates
MIPMAPPEDIMAGE	The image is smoothed when scaled down to less than its original size

Take a look at the screenshot taken from the program below. One image was loaded with flag=0 and the other with the FILTEREDIMAGE flag set. One is less pixelated than the other when scaled 3X.

```
Graphics 640,480
Local URL:String="http::www.2dgamecreators.com/tutorials/gameprogramming
/basic/"
Local Player:TImage=LoadImage(LoadBank(URL+"blobship_1-1.png"),0)
Local
Player1:TImage=LoadImage(LoadBank(URL+"blobship_1-1.png"),FILTEREDIMAGE)

Repeat
  Cls
  SetScale 3.0,3.0
  DrawImage Player1,MouseX(),MouseY()
  DrawImage Player,MouseX()+200,MouseY()
  Flip
Until KeyHit(key_escape) Or AppTerminate()
End
```

If you need to combine the flags you can use the | operator as follows:-

```
Local
Alien:TImage=LoadImage("cartoonufo_1-1.png",FILTEREDIMAGE|MIPMAPPEDIMAGE)
```

Try and experiment with the other flag and also the combined flag to see what happens.

Summary

In this tutorial we saw how to transform our images before drawing them to the graphic screen and how the transformed images are affected by the flags we use during image loading.

In the next tutorial, we are going to about blending modes and how it affect the images that we draw on the graphic screen..

Back to the index, Next Tutorial