

MBS Real Studio LargePicture Plugin Documentation

Christian Schmitz

May 15, 2012

0.1 Introduction

This is the PDF version of the documentation for the Real Studio Plug-in from Monkeybread Software Germany. Plugin part: MBS Real Studio LargePicture Plugin

0.2 Content

- 1 List of all topics 3
- 2 All items in this plugin 11
- 3 List of all classes 135

Chapter 1

List of Topics

• 2 Large Picture	11
– 2.1 class PictureMBS	11
* 2.1.1 AlphaChannel as PictureMBS	12
* 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer) as PictureMBS	13
* 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer, delta as integer) as PictureMBS	14
* 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer, delta as integer, ScaleFactor as double) as PictureMBS	15
* 2.1.1 AutoLevel as boolean	17
* 2.1.1 AutoLevel(x as integer, y as integer, w as integer, h as integer) as boolean	17
* 2.1.1 BlackChannel as PictureMBS	18
* 2.1.1 BlueChannel as PictureMBS	18
* 2.1.1 BoxBlurFilter(dest as PictureMBS, Radius as double, Iterations as integer, Vertical as boolean = true, Horizontal as boolean = true) as PictureMBS	19
* 2.1.1 BoxBlurFilter(dest as PictureMBS, Radius as double, Vertical as boolean = true, Horizontal as boolean = true) as PictureMBS	19
* 2.1.1 BoxBlurFractionalFilter(dest as PictureMBS, Radius as double) as PictureMBS	20
* 2.1.1 CalculateMemory(width as integer, height as integer, theImageFormat as integer) as Int64	20
* 2.1.1 CanAllocateImage(width as integer, height as integer, theImageFormat as integer) as boolean	21
* 2.1.1 Channel(index as integer) as PictureMBS	21
* 2.1.1 ClearRect	22
* 2.1.1 ClearRect(x as integer, y as integer, width as integer, height as integer)	22
* 2.1.1 ClipImage as PictureMBS	23

* 2.1.1 ClipImage(x as integer, y as integer, width as integer, height as integer) as PictureMBS	23
* 2.1.1 Clone as PictureMBS	23
* 2.1.1 Close	23
* 2.1.1 CMYKChannels as PictureMBS	24
* 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean	24
* 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean	26
* 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean	28
* 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean	30
* 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean	32
* 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean	34
* 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean	37
* 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean	39
* 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean	41
* 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean	43
* 2.1.1 CompareImages(other as PictureMBS) as Int64	45
* 2.1.1 Constructor(Buf as MemoryBlock, width as integer, height as integer, ImageFormat as integer, RowSize as integer)	46
* 2.1.1 Constructor(pic as picture, UseAlpha as boolean=false)	46
* 2.1.1 Constructor(width as integer, height as integer, ImageFormat as integer)	47
* 2.1.1 Constructor(width as integer, height as integer, ImageFormat as integer, BlockSize as integer, FilePath as folderitem)	48
* 2.1.1 CopyGWorld as variant	49

* 2.1.1 CopyMask as picture	49
* 2.1.1 CopyMask(x as integer, y as integer, w as integer, h as integer) as picture	50
* 2.1.1 CopyPicture as picture	50
* 2.1.1 CopyPicture(x as integer, y as integer, w as integer, h as integer) as picture	51
* 2.1.1 CopyPictureWithMask as picture	51
* 2.1.1 CopyPictureWithMask(x as integer, y as integer, w as integer, h as integer) as picture	52
* 2.1.1 CopyPixels(source as PictureMBS) as boolean	52
* 2.1.1 CopyPixels(source as PictureMBS, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) as boolean	53
* 2.1.1 CopyPixels(source as PictureMBS, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) as boolean	54
* 2.1.1 CreatePictureMBS(width as integer, height as integer, ImageFormat as integer) as PictureMBS	55
* 2.1.1 CreatePictureMBS(width as integer, height as integer, theImageFormat as integer) as PictureMBS	55
* 2.1.1 CyanChannel as PictureMBS	56
* 2.1.1 DataStringInFormat(ImageFormat as integer) as string	56
* 2.1.1 DiffuseFilter(dest as PictureMBS, level as integer) as PictureMBS	56
* 2.1.1 DitherFilter(dest as PictureMBS, matrix as integer, levels as integer) as PictureMBS	57
* 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False)	57
* 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False)	58
* 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, InvertMask as boolean=False)	59
* 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, InvertMask as boolean=False)	59
* 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False)	60
* 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False)	61
* 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, InvertMask as boolean=False)	61
* 2.1.1 DrawMaskedPictureRGB(pic as picture, InvertMask as boolean=False)	62
* 2.1.1 DrawPictureBlueToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer)	63
* 2.1.1 DrawPictureBlueToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer)	63
* 2.1.1 DrawPictureGreenToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer)	63
* 2.1.1 DrawPictureGreenToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer)	64

* 2.1.1 DrawPictureRedToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer)	64
* 2.1.1 DrawPictureRedToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer)	64
* 2.1.1 DrawPictureRGB(pic as picture)	65
* 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer)	65
* 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer)	66
* 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer)	67
* 2.1.1 EngraveFilter(dest as PictureMBS, level as integer) as PictureMBS	67
* 2.1.1 FillRect(value as integer)	68
* 2.1.1 FillRect(x as integer, y as integer, width as integer, height as integer, value as integer)	68
* 2.1.1 FillRectRandom	69
* 2.1.1 FillRectRandom(x as integer, y as integer, width as integer, height as integer)	69
* 2.1.1 FillRectRGB(FillColor as color)	70
* 2.1.1 FillRectRGB(FillColor as color, alpha as integer)	70
* 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer)	71
* 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer)	72
* 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color)	73
* 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer)	73
* 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer)	74
* 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer)	75
* 2.1.1 GainFilter(dest as PictureMBS, gain as double, bias as double) as PictureMBS	76
* 2.1.1 GammaFilter(dest as PictureMBS, gamma as double) as PictureMBS	76
* 2.1.1 GammaFilter(dest as PictureMBS, gamma as double, alphaGamma as double) as PictureMBS	76
* 2.1.1 GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double) as PictureMBS	77
* 2.1.1 GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double, alphaGamma as double) as PictureMBS	78
* 2.1.1 GrayChannel as PictureMBS	78
* 2.1.1 GreenChannel as PictureMBS	79
* 2.1.1 HMirror	79
* 2.1.1 Invert	80
* 2.1.1 Invert(x as integer, y as integer, w as integer, h as integer)	80
* 2.1.1 MagentaChannel as PictureMBS	80
* 2.1.1 MirroredView as PictureMBS	80

* 2.1.1 NeonFilter(dest as PictureMBS) as PictureMBS	81
* 2.1.1 OilFilter(dest as PictureMBS, levels as integer, range as integer) as PictureMBS	81
* 2.1.1 RowRow(index as integer) as memoryblock	82
* 2.1.1 RedChannel as PictureMBS	82
* 2.1.1 RGBChannels as PictureMBS	83
* 2.1.1 Rotate(angle as double, Red as integer = 0, Green as integer = 0, Blue as integer = 0, Alpha as integer = 0, Gray as integer = 0, Cyan as integer = 0, Magenta as integer = 0, Yellow as integer = 0, Black as integer = 0) as PictureMBS	84
* 2.1.1 Rotate180	84
* 2.1.1 Rotate180(dest as PictureMBS=nil) as PictureMBS	85
* 2.1.1 Rotate270(dest as PictureMBS=nil) as PictureMBS	85
* 2.1.1 Rotate270slow(dest as PictureMBS=nil) as PictureMBS	86
* 2.1.1 Rotate90(dest as PictureMBS=nil) as PictureMBS	86
* 2.1.1 Rotate90slow(dest as PictureMBS=nil) as PictureMBS	87
* 2.1.1 Row(index as integer) as memoryblock	87
* 2.1.1 RowInFormat(index as integer, ImageFormat as integer) as memoryblock	88
* 2.1.1 RowInFormat(index as integer, ImageFormat as integer, InvertAlpha as boolean) as memoryblock	89
* 2.1.1 RowStringInFormat(index as integer, ImageFormat as integer) as string	89
* 2.1.1 Scale(source as PictureMBS, temp as PictureMBS, mode as integer, width as integer, height as integer) as boolean	90
* 2.1.1 ScaleFast(source as PictureMBS, width as integer, height as integer) as boolean	91
* 2.1.1 ScaleMT(threads as integer, source as PictureMBS, temp as PictureMBS, mode as integer, width as integer, height as integer) as boolean	91
* 2.1.1 SolarizeFilter(dest as PictureMBS) as PictureMBS	92
* 2.1.1 StampFilter(dest as PictureMBS, radius as double, threshold as double, softness as double, Black as Color, White as Color) as PictureMBS	92
* 2.1.1 TransferFilter(dest as PictureMBS, gray() as integer) as PictureMBS	93
* 2.1.1 TransferFilter(dest as PictureMBS, gray() as integer, alpha() as integer) as PictureMBS	94
* 2.1.1 TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer) as PictureMBS	94
* 2.1.1 TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer, alpha() as integer) as PictureMBS	95
* 2.1.1 UnsharpFilter(origpixels as PictureMBS, Amount as double, Threshold as integer) as boolean	96
* 2.1.1 VMirror	96
* 2.1.1 YellowChannel as PictureMBS	97
* 2.1.2 AlphaOffset as Integer	97
* 2.1.2 BlackOffset as Integer	98
* 2.1.2 BlueOffset as Integer	98
* 2.1.2 ChannelCount as Integer	98
* 2.1.2 CyanOffset as Integer	99

* 2.1.2 DebugPicture as Picture	99
* 2.1.2 DebugPictureEnabled as Boolean	99
* 2.1.2 Factory as PictureFactoryMBS	100
* 2.1.2 GrayOffset as Integer	100
* 2.1.2 GreenOffset as Integer	100
* 2.1.2 HasAlpha as Boolean	101
* 2.1.2 HasBlack as Boolean	101
* 2.1.2 HasBlue as Boolean	101
* 2.1.2 HasCyan as Boolean	102
* 2.1.2 HasGray as Boolean	102
* 2.1.2 HasGreen as Boolean	102
* 2.1.2 HasMagenta as Boolean	103
* 2.1.2 HasRed as Boolean	103
* 2.1.2 HasYellow as Boolean	103
* 2.1.2 Height as Integer	104
* 2.1.2 ImageFormat as Integer	104
* 2.1.2 ImageFormatString as String	105
* 2.1.2 IsCMYK as Boolean	105
* 2.1.2 IsGray as Boolean	105
* 2.1.2 IsMapping as Boolean	106
* 2.1.2 IsRGB as Boolean	106
* 2.1.2 MagentaOffset as Integer	107
* 2.1.2 MappingBlockSize as Integer	107
* 2.1.2 Memory as Memoryblock	107
* 2.1.2 MemoryTarget as Memoryblock	107
* 2.1.2 Parent as PictureMBS	108
* 2.1.2 PixelSize as Integer	108
* 2.1.2 RedOffset as Integer	108
* 2.1.2 RowOffset as Integer	109
* 2.1.2 RowSize as Integer	109
* 2.1.2 Target as Picture	110
* 2.1.2 TotalSize as Int64	110
* 2.1.2 Valid as Boolean	111
* 2.1.2 Width as Integer	111
* 2.1.2 YellowOffset as Integer	111
* 2.1.2 YieldTicks as Integer	112
* 2.1.3 Dither90Halftone6x6Matrix = 5	112
* 2.1.3 DitherCluster3Matrix = 8	112
* 2.1.3 DitherCluster4Matrix = 9	113
* 2.1.3 DitherCluster8Matrix = & h0000000A	113
* 2.1.3 DitherLines4x4Matrix = 4	113
* 2.1.3 DitherMagic2x2Matrix = 1	113

* 2.1.3 DitherMagic4x4Matrix = 2	113
* 2.1.3 DitherOrdered4x4Matrix = 3	113
* 2.1.3 DitherOrdered6x6Matrix = 6	114
* 2.1.3 DitherOrdered8x8Matrix = 7	114
* 2.1.3 ImageFormat1of3 = & h0000000F	114
* 2.1.3 ImageFormat1of4 = & h00000012	114
* 2.1.3 ImageFormat2of3 = & h00000010	114
* 2.1.3 ImageFormat2of4 = & h00000013	115
* 2.1.3 ImageFormat3of3 = & h00000011	115
* 2.1.3 ImageFormat3of4 = & h00000014	115
* 2.1.3 ImageFormat4of4 = & h00000015	115
* 2.1.3 ImageFormatABGR = 9	116
* 2.1.3 ImageFormatACMYK = & h00000019	116
* 2.1.3 ImageFormatAG = & h0000000D	116
* 2.1.3 ImageFormatAKYMC = & h0000001E	116
* 2.1.3 ImageFormatAofABGR = & h00000012	117
* 2.1.3 ImageFormatAofARGB = & h00000012	117
* 2.1.3 ImageFormatAofBGRA = & h00000015	117
* 2.1.3 ImageFormatAofRGBA = & h00000015	118
* 2.1.3 ImageFormatARGB = 4	118
* 2.1.3 ImageFormatBGR = 6	118
* 2.1.3 ImageFormatBGRA = 7	118
* 2.1.3 ImageFormatBGRX = 8	119
* 2.1.3 ImageFormatBofABGR = & h00000013	119
* 2.1.3 ImageFormatBofARGB = & h00000015	119
* 2.1.3 ImageFormatBofBGR = & h0000000F	120
* 2.1.3 ImageFormatBofBGRA = & h00000012	120
* 2.1.3 ImageFormatBofRGB = & h00000011	120
* 2.1.3 ImageFormatBofRGBA = & h00000014	121
* 2.1.3 ImageFormatBuffer = & h00000016	121
* 2.1.3 ImageFormatCMYK = & h00000017	121
* 2.1.3 ImageFormatCMYKA = & h00000018	121
* 2.1.3 ImageFormatCMYKX = & h0000001A	122
* 2.1.3 ImageFormatG = & h0000000B	122
* 2.1.3 ImageFormatGA = & h0000000C	122
* 2.1.3 ImageFormatGofABGR = & h00000014	122
* 2.1.3 ImageFormatGofARGB = & h00000014	123
* 2.1.3 ImageFormatGofBGR = & h00000010	123
* 2.1.3 ImageFormatGofBGRA = & h00000013	123
* 2.1.3 ImageFormatGofRGB = & h00000010	124
* 2.1.3 ImageFormatGofRGBA = & h00000013	124
* 2.1.3 ImageFormatKYMC = & h0000001C	124

* 2.1.3 ImageFormatKYMCA = & h0000001D	125
* 2.1.3 ImageFormatKYMCA = & h0000001F	125
* 2.1.3 ImageFormatRGB = 1	125
* 2.1.3 ImageFormatRGBA = 2	125
* 2.1.3 ImageFormatRGBX = 3	126
* 2.1.3 ImageFormatRofABGR = & h00000015	126
* 2.1.3 ImageFormatRofARGB = & h00000013	127
* 2.1.3 ImageFormatRofBGR = & h00000011	127
* 2.1.3 ImageFormatRofBGRA = & h00000014	127
* 2.1.3 ImageFormatRofRGB = & h0000000F	128
* 2.1.3 ImageFormatRofRGBA = & h00000012	128
* 2.1.3 ImageFormatScaling1 = & h00000021	128
* 2.1.3 ImageFormatScaling2 = & h00000022	129
* 2.1.3 ImageFormatScaling3 = & h00000023	129
* 2.1.3 ImageFormatScaling4 = & h00000024	129
* 2.1.3 ImageFormatScaling5 = & h00000025	129
* 2.1.3 ImageFormatScaling6 = & h00000026	130
* 2.1.3 ImageFormatScaling7 = & h00000027	130
* 2.1.3 ImageFormatScaling8 = & h00000028	130
* 2.1.3 ImageFormatUnknown = 0	130
* 2.1.3 ImageFormatXBGR = & h0000000A	131
* 2.1.3 ImageFormatXCMYK = & h0000001B	131
* 2.1.3 ImageFormatXKYMC = & h00000020	131
* 2.1.3 ImageFormatXRGB = 5	131
* 2.1.3 ScaleBox = 2	132
* 2.1.3 ScaleCubic = 7	132
* 2.1.3 ScaleLanczos3 = 3	132
* 2.1.3 ScaleLanczos8 = 4	132
* 2.1.3 ScaleMitchell = 5	132
* 2.1.3 ScalePoly3 = 6	132
* 2.1.3 ScaleTriangle = 1	133
– 2.2 class PictureFactoryMBS	133
* 2.2.1 SetFactory(factory as PictureFactoryMBS)	133
* 2.2.2 NewPictureMBS(Width as integer, Height as integer, ImageFormat as integer) as PictureMBS	133

Chapter 2

Large Picture

2.1 class PictureMBS

class PictureMBS

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The MBS picture class for really large pictures.

Example:

```
dim fSource as FolderItem = SpecialFolder.Desktop.Child("test.png") // some png with alpha
dim oPNGInput as new PNGReaderMBS

If oPNGInput.OpenFile(fSource) Then
If oPNGInput.ApplyOptions(0) Then

dim imgSource as New PictureMBS(oPNGInput.Width, oPNGInput.Height, PictureMBS.ImageFormatRGBA)

' Read row by row the file and puts it in a PictureMBS instance

dim nMax as integer = oPNGInput.Height - 1
For nInd as integer = 0 To nMax
imgSource.RowInFormat(nInd, PictureMBS.ImageFormatRGBA, true) = oPNGInput.ReadRow()
Next

' show only alpha/mask channel
Backdrop=imgSource.AlphaChannel.CopyPicture

' show Picture without mask
```

```

Backdrop=imgSource.CopyPicture
' show picture with mask
Backdrop=imgSource.CopyPictureWithMask

End If
End If

```

Notes:

Using virtual memory you are only limited to hard disc space for swapping.

The REALbasic picture class is limited to 2 GB and to width/height being in platform specific ranges. This class works with pictures up to 100 million pixels width and 2 billion pixels height.

2.1.1 Methods**AlphaChannel as PictureMBS**

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The alpha channel as a new PictureMBS object.

Example:

```

dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
dim r as PictureMBS = p.AlphaChannel
r.fillrect(100) // fill only alpha channel

```

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer) as PictureMBS

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies a 5x5 matrix to the picture.

Example:

```
dim matrix(24) as integer
dim x,y as integer
dim value as integer

for y=0 to 4
for x=0 to 4
matrix(x+y*5)=value // fill matrix
next
next

dim s,d as PictureMBS // make source and dest somewhere

d=s.ApplyMatrix(d, 5, matrix)
```

Notes:

MatrixDimension: Size of the matrix: 1 to 50. This is the width and height of the matrix.

matrix: The matrix array must contain exactly MatrixDimension*MatrixDimension values. (ubound(matrix)=MatrixDimension-1)

delta: Optional value. Default is 0.

ScaleFactor: Optional value. Default is 1.0.

if dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.

If dest is not nil, it must match the size of the original picture.

For each pixel in the dest image the following operation is done:

Make sum of all source pixels multiplied with their matrix entry.

add to the sum the delta value

multiply the sum by ScaleFactor

See the example project for several example matrices.

A matrix value of 255 or more leaves the dest pixel away from the sum.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer, delta as integer) as PictureMBS 14
- 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer, delta as integer, ScaleFactor as double) as PictureMBS 15

ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer, delta as integer) as PictureMBS

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies a 5x5 matrix to the picture.

Example:

```
dim matrix(24) as integer
dim x,y as integer
dim value as integer

for y=0 to 4
for x=0 to 4
matrix(x+y*5)=value // fill matrix
next
next

dim s,d as PictureMBS // make source and dest somewhere

d=s.ApplyMatrix(d, 5, matrix, 5)
```

Notes:

MatrixDimension: Size of the matrix: 1 to 50. This is the width and height of the matrix.

matrix: The matrix array must contain exactly MatrixDimension*MatrixDimension values. (ubound(matrix)=MatrixDimension-1)

delta: Optional value. Default is 0.

ScaleFactor: Optional value. Default is 1.0.

if dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.
If dest is not nil, it must match the size of the original picture.

For each pixel in the dest image the following operation is done:
Make sum of all source pixels multiplied with their matrix entry.
add to the sum the delta value
multiply the sum by ScaleFactor

See the example project for several example matrices.

A matrix value of 255 or more leaves the dest pixel away from the sum.

Works with Gray, RGB and CMYK pictures and supports alpha channel.
See also:

- 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer) as PictureMBS 13
- 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer, delta as integer, ScaleFactor as double) as PictureMBS 15

ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer, delta as integer, ScaleFactor as double) as PictureMBS

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies a 5x5 matrix to the picture.

Example:

```
dim matrix(24) as integer
dim x,y as integer
dim value as integer
```

```
for y=0 to 4
for x=0 to 4
matrix(x+y*5)=1 // fill matrix
next
next
```

```
dim s,d as PictureMBS // make source and dest somewhere
```

```
s = new PictureMBS(LogoMBS(500))
```

```
d = new PictureMBS(500, 500, PictureMBS.ImageFormatRGB)

// Blur with 5x5 Matrix
d=s.ApplyMatrix(d, 5, matrix, 1, 1.0/25.0)

Backdrop = d.CopyPicture
```

Notes:

MatrixDimension: Size of the matrix: 1 to 50. This is the width and height of the matrix.

matrix: The matrix array must contain exactly MatrixDimension*MatrixDimension values. (ubound(matrix)=MatrixDimension-1)

delta: Optional value. Default is 0.

ScaleFactor: Optional value. Default is 1.0.

if dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.

If dest is not nil, it must match the size of the original picture.

For each pixel in the dest image the following operation is done:

Make sum of all source pixels multiplied with their matrix entry.

add to the sum the delta value

multiply the sum by ScaleFactor

See the example project for several example matrices.

A matrix value of 255 or more leaves the dest pixel away from the sum.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer) as PictureMBS 13
- 2.1.1 ApplyMatrix(dest as PictureMBS, MatrixDimension as integer, matrix() as integer, delta as integer) as PictureMBS 14

AutoLevel as boolean

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies auto levels on the picture.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
call p.AutoLevel
window1.Backdrop = p.CopyPicture
```

Notes:

The histogram is built, white and black points are searched and all pixels adjusted.
Returns true on success and false on any error.

Works only with RGB pictures.

See also:

- 2.1.1 AutoLevel(x as integer, y as integer, w as integer, h as integer) as boolean

17

AutoLevel(x as integer, y as integer, w as integer, h as integer) as boolean

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies auto levels on the picture.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
call p.AutoLevel(0,0,50,50)
window1.Backdrop = p.CopyPicture
```

Notes:

The histogram is built, white and black points are searched and all pixels adjusted.
Returns true on success and false on any error.

Works only with RGB pictures.
See also:

- 2.1.1 AutoLevel as boolean

17

BlackChannel as PictureMBS

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The black channel of a CMYK picture as a new PictureMBS object.

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

BlueChannel as PictureMBS

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The blue channel as a new PictureMBS object.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
dim r as PictureMBS = p.BlueChannel
r.fillrect(100) // fill only blue channel
```

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

BoxBlurFilter(dest as PictureMBS, Radius as double, Iterations as integer, Vertical as boolean = true, Horizontal as boolean = true) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The box blur filter.
Example:

```
Dim boxPic,tempObj As PictureMBS
dim logo as Picture = LogoMBS(500)
dim pictureObj as new PictureMBS(logo)
```

```
tempObj = New PictureMBS(pictureObj.Width, pictureObj.Height, pictureObj.ImageFormat)
boxPic = pictureObj.BoxBlurFilter(tempObj, 3.0, 3)
```

```
Backdrop=boxpic.CopyPicture
```

Notes:

if dest is nil, the picture factory is used to create a new picture.
 On success dest or the new picture is returned.
 If dest is not nil, it must match the size of the original picture.

Vertical and Horizontal define whether effect is applied horizontal and/or vertical.

Returns nil on any error.
 Works with Gray, RGB and CMYK pictures and supports alpha channel.
 See also:

- 2.1.1 BoxBlurFilter(dest as PictureMBS, Radius as double, Vertical as boolean = true, Horizontal as boolean = true) as PictureMBS 19

BoxBlurFilter(dest as PictureMBS, Radius as double, Vertical as boolean = true, Horizontal as boolean = true) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The box blur filter.
Notes:

if dest is nil, the picture factory is used to create a new picture.
 On success dest or the new picture is returned.
 If dest is not nil, it must match the size of the original picture.

Vertical and Horizontal define whether effect is applied horizontal and/or vertical.

Returns nil on any error.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 BoxBlurFilter(dest as PictureMBS, Radius as double, Iterations as integer, Vertical as boolean = true, Horizontal as boolean = true) as PictureMBS 19

BoxBlurFractionalFilter(dest as PictureMBS, Radius as double) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The box blur filter for the radius fraction.

Notes:

If you call BoxBlurFilter and BoxBlurFractionalFilter with a radius of 3.5 the BoxBlurFilter does the 3.0 and BoxBlurFractionalFilter does the 0.5.

if dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.

If dest is not nil, it must match the size of the original picture.

Returns nil on any error.

CalculateMemory(width as integer, height as integer, theImageFormat as integer) as Int64

Plugin Version: 10.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calculates the memory needed for allocating the image.

Example:

```
dim n as int64 = PictureMBS.CalculateMemory(1000, 1000, PictureMBS.ImageFormatRGB)
MsgBox str(n)
```

Notes: Returns number of bytes needed.

CanAllocateImage(width as integer, height as integer, theImageFormat as integer) as boolean

Plugin Version: 10.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Can the image with this size be allocated.

Example:

```
dim n as Boolean = PictureMBS.CanAllocateImage(1000, 1000, PictureMBS.ImageFormatRGB)
MsgBox str(n)
dim x as Boolean = PictureMBS.CanAllocateImage(100000, 100000, PictureMBS.ImageFormatRGB)
MsgBox str(x)
```

Notes: Returns true if possible and false if the size is too big.

Channel(index as integer) as PictureMBS

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the channel with the given index as a new picture object.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
dim r as PictureMBS = p.Channel(0)
r.fillrect(100) // fill only red channel
```

Notes:

Returns nil on any error. May raise an out of bounds exception on invalid index. Index is zero based. No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture. Use this function to access the pixels of the channel directly. The resulting PictureMBS object is a grayscale picture.

ClearRect

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Clears all pixels.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.ClearRect
window1.Backdrop = p.CopyPicture
```

Notes:

Writes zeros over all pixels and all channels.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 ClearRect(x as integer, y as integer, width as integer, height as integer)

22

ClearRect(x as integer, y as integer, width as integer, height as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Clears all pixels in the given area.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.ClearRect(0,0,100,100)
window1.Backdrop = p.CopyPicture
```

Notes:

Writes zeros over all pixels and all channels.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 ClearRect

22

ClipImage as PictureMBS

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a new PictureMBS object for the same picture content.

Notes: This may be useful if you need a second PictureMBS object. For example if two threads work on different rows.

See also:

- 2.1.1 ClipImage(x as integer, y as integer, width as integer, height as integer) as PictureMBS 23

ClipImage(x as integer, y as integer, width as integer, height as integer) as PictureMBS

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a new PictureMBS object which draws only into a portion of the existing image.

Notes: This may be useful to apply an effect only on a portion of an existing image.

See also:

- 2.1.1 ClipImage as PictureMBS 23

Clone as PictureMBS

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a copy of a picture.

Notes:

Does not work for pictures using virtual memory.

(Fails if IsMapping=True)

Copies the whole picture even if you clone just one channel.

Returns nil on low memory.

Close

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Closes the picture by releasing all memory.

Notes: This calls the destructor internally.

CMYKChannels as PictureMBS

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The CMYK channels as a new PictureMBS object.

Notes:

Returns nil if the image is not a CMYK picture.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the CMYK pixels directly without modifying an alpha channel

The resulting PictureMBS object is a CMYK picture.

Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Example:

```
dim DestImage As PictureMBS
dim Image As PictureMBS
dim Mask As PictureMBS
dim DestX As Integer=100
dim DestY As Integer=100
dim SourceX As Integer=0
dim SourceY As Integer=0
dim Width As Integer=500
dim Height As Integer=500
```

```
image=new PictureMBS(LogoMBS(500))
Mask=nil
DestImage=new PictureMBS(700,700,PictureMBS.ImageFormatRGB)
```

```
// this will only copy the pixels
if DestImage.Combine(image,Mask, DestX, DestY, SourceX, SourceY, Width, Height, false) then
window1.Backdrop=DestImage.CopyPicture
end if
```

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the fore color applying the mask.
4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean

- 2.1.1 `Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color)` as boolean 28
- 2.1.1 `Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer)` as boolean 30
- 2.1.1 `Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer)` as boolean 32
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean)` as boolean 34
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color)` as boolean 37
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color)` as boolean 39
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer)` as boolean 41
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer)` as boolean 43

Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.

2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the the pixels are filled with the fore color applying the mask.
4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 24
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 28
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 30
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer,

- SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 32
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 34
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 37
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 39
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 41
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 43

Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Example:

```
dim DestImage As PictureMBS
dim Image As PictureMBS
dim Mask As PictureMBS
dim DestX As Integer=100
dim DestY As Integer=100
dim SourceX As Integer=0
dim SourceY As Integer=0
dim Width As Integer=500
dim Height As Integer=500
```

```
image=new PictureMBS(LogoMBS(500))
Mask=nil
DestImage=new PictureMBS(700,700,PictureMBS.ImageFormatRGB)
```

```
if DestImage.Combine(image,Mask, DestX, DestY, SourceX, SourceY, Width, Height, true, & cFF0000, & cFF0000)
then
window1.Backdrop=DestImage.CopyPicture
```

end if

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the fore color applying the mask.
4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha

channels. To make sure the alpha channel is not touched, use the `PictureMBS.RGBChannels` function and pass that new `PictureMBS`.

See also:

- 2.1.1 `Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean` 24
- 2.1.1 `Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean` 26
- 2.1.1 `Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean` 30
- 2.1.1 `Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean` 32
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean` 34
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean` 37
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean` 39
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean` 41
- 2.1.1 `Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean` 43

`Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean`

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the fore color applying the mask.
4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean)

- as boolean 24
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 26
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 28
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 32
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 34
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 37
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 39
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 41
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 43

Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the fore color applying the mask.
4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 24
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 26
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer,

- SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 28
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 30
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 34
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 37
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 39
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 41
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 43

Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Example:

```

dim DestImage As PictureMBS
dim Image As PictureMBS
dim Mask As PictureMBS
dim DestX As Integer=100
dim DestY As Integer=100
dim SourceX As Integer=0
dim SourceY As Integer=0
dim Width As Integer=500
dim Height As Integer=500

// we create a little mask for a smooth fade
dim m as Picture = NewPicture(500,500,32)
dim g as Graphics = m.Graphics

```

```

for y as integer = 0 to 499
dim n as integer = y*255/499
g.ForeColor = rgb(n, n, n)
g.DrawLine 0,y,499,y
next

// uncomment to see our mask:
'Backdrop = m
'return

image=new PictureMBS(LogoMBS(500))
Mask=new PictureMBS(m)
DestImage=new PictureMBS(700,700,PictureMBS.ImageFormatRGB)

// this will only copy the pixels
if DestImage.Combine(image,false,Mask,DestX,DestY,SourceX,SourceY,Width,Height,false) then
window1.Backdrop=DestImage.CopyPicture
end if

```

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the fore color applying the mask.
4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 24
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 26
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 28
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 30
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 32
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 37
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 39

- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 41
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 43

Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the foreground color applying the mask.
4. As the last variation the pixels are copied and the foreground color, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 24
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 26
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 28
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 30
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 32
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 34
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 39
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 41
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 43

Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the fore color applying the mask.
4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either

integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 24
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 26
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 28
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 30
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 32
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 34
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 37
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 41
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 43

Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Example:

```
dim DestImage As PictureMBS
dim Image As PictureMBS
dim Mask As PictureMBS
dim DestX As Integer=100
dim DestY As Integer=100
dim SourceX As Integer=0
dim SourceY As Integer=0
dim Width As Integer=500
dim Height As Integer=500
dim UseColours as Boolean = false
dim ForeColour as color = & cFF0000

image=new PictureMBS(LogoMBS(500))
Mask=nil
DestImage=new PictureMBS(700,700,PictureMBS.ImageFormatRGB)

if DestImage.Combine(image,Mask, DestX, Dest Y, SourceX, SourceY, Width, Height, UseColours, ForeColour) then
window1.Backdrop=DestImage.CopyPicture
end if
```

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the fore color applying the mask.

4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 24
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 26
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 28
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 30
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean,

- ForeColour As Integer, MaskColour As Integer) as boolean 32
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 34
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 37
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 39
 - 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 43

Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from one picture into another picture with some options.

Notes:

Returns true on success and false on failure.

This function has 4 behaviors depending on the parameters:

1. If mask is nil and no ForeColour and MaskColour values are passed, the pixels are copied to the destination picture.
2. But if there is a mask, the pixels are copied with applying the mask.
3. If the mask color is not defined, the pixels are filled with the fore color applying the mask.
4. As the last variation the pixels are copied and the forecolor, the mask color or black is used with the image as the mask. If UseColours parameter is false black is used for this.

Parameters:

Image: the source picture, must not be nil.

PreMultipliedSource: Optional parameter. If true the image must be premultiplied. Default is false.

Mask: the mask picture, can be nil.

DestX: destination position

DestY: destination position

SourceX: source position

SourceY: source position

Width: width of the area to copy

Height: height of the area to copy

UseColours: whether to use the mask colour.

ForeColour: the fore colour, optional, can be integer or color

MaskColour: the mask color, optional, can be integer or color

This function is 5 times in the plugin defined to implement having the last two parameters optional and either integer or color. You can pass a negative number for MaskColour or ForeColour to disable this parameter.

The images you use can be Gray, RGB with or without alpha channels. But most variants here ignore alpha channels. To make sure the alpha channel is not touched, use the PictureMBS.RGBChannels function and pass that new PictureMBS.

See also:

- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 24
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 26
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 28
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 30
- 2.1.1 Combine(Image As PictureMBS, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer, MaskColour As Integer) as boolean 32
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean) as boolean 34

- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color) as boolean 37
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As color, MaskColour As color) as boolean 39
- 2.1.1 Combine(Image As PictureMBS, PreMultipliedSource as boolean, Mask As PictureMBS, DestX As Integer, DestY As Integer, SourceX As Integer, SourceY As Integer, Width As Integer, Height As Integer, UseColours As Boolean, ForeColour As Integer) as boolean 41

CompareImages(other as PictureMBS) as Int64

Plugin Version: 10.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Compares two pictures.

Example:

```
dim p as new PictureMBS(1000,1000, PictureMBS.imageFormatRGB)

// fill random
p.FillRectRandom

dim q as new PictureMBS(1000,1000, PictureMBS.imageFormatRGB)

// copy pixels
call q.CopyPixels(p, 0, 0, 1000, 1000, 0, 0)
q.FillRect(0,0,10,10,0) // fill 100 pixels

// show image
Backdrop = q.CopyPicture

// and compare
Title = str(p.CompareImages(q)) // shows 100
```

Notes:

Returns -1 if both pictures are not from the same structure. (e.g. compare gray with RGB)
Else returns the number of different pixels.

Constructor(Buf as MemoryBlock, width as integer, height as integer, ImageFormat as integer, RowSize as integer)

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a PictureMBS object based on a memoryblock.

See also:

- 2.1.1 Constructor(pic as picture, UseAlpha as boolean=false) 46
- 2.1.1 Constructor(width as integer, height as integer, ImageFormat as integer) 47
- 2.1.1 Constructor(width as integer, height as integer, ImageFormat as integer, BlockSize as integer, FilePath as folderitem) 48

Constructor(pic as picture, UseAlpha as boolean=false)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: No, Linux: No, . **Function:** Creates a PictureMBS which shares memory with the given picture.

Example:

```
// Create a picture with mask:
dim p as Picture = LogoMBS(200)
dim g as Graphics = p.mask.Graphics

g.ForeColor = & cFFFFFF
g.FillRect 0,0,g.Width,g.Height

g.ForeColor = & c000000
g.Filloval 0,0,g.Width,g.Height

canvas1.Backdrop = p

// create PictureMBS
dim pic as new PictureMBS(p, true)
dim mask as new PictureMBS(p.mask)

// draw mask into alpha channel
call pic.AlphaChannel.CopyPixels(mask,0,0,mask.Width,mask.Height,0,0)

// and copy back to REALbasic picture
canvas2.Backdrop = pic.CopyPictureWithMask
```

Notes:

All drawings in the Picture and in the PictureMBS object will be visible in both objects.
 This function works on Mac OS and Windows with both 24 bit and 32 bit pictures.
 On Mac this function can fail if the picture is not a GWorld (Bitmap) picture.

The Valid property is set to true on success.

If you set UseAlpha=True, the 4th channel in a 32 bit picture is available for you as an alpha channel. REALbasic does not use the 4th channel in the picture data and 24 bit pictures do not have one. So you can use 32 bit pictures, copy the pictures mask in the alpha channel (using PictureMBS.AlphaChannel. inverting may be needed), perform some operations and later make a copy of the of the image to a picture and extract the alpha channel back into the pictue's mask.

Added support for Console/Web targets in 12.2 plugins. Please be aware that alpha channel of pictures with alpha channel has only a range from 0 to 127 for the values.

See also:

- 2.1.1 Constructor(Buf as MemoryBlock, width as integer, height as integer, ImageFormat as integer, RowSize as integer) 46
- 2.1.1 Constructor(width as integer, height as integer, ImageFormat as integer) 47
- 2.1.1 Constructor(width as integer, height as integer, ImageFormat as integer, BlockSize as integer, FilePath as folderitem) 48

Constructor(width as integer, height as integer, ImageFormat as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a new PictureMBS object with the given size and Imageformat.

Notes:

ImageFormat must be one of the ImageFormat constants.
 The Valid property is set to true on success.

The constructor allocated address space for the image.
 Physical memory is allocated based on write access to pixels.
 See also:

- 2.1.1 Constructor(Buf as MemoryBlock, width as integer, height as integer, ImageFormat as integer, RowSize as integer) 46
- 2.1.1 Constructor(pic as picture, UseAlpha as boolean=false) 46

- 2.1.1 Constructor(width as integer, height as integer, ImageFormat as integer, BlockSize as integer, FilePath as folderitem) 48

Constructor(width as integer, height as integer, ImageFormat as integer, BlockSize as integer, FilePath as folderitem)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a PictureMBS object using virtual memory.

Example:

```
dim f as folderitem = SpecialFolder.Desktop.child("tempdata")
dim p as new PictureMBS(100000, 100000, PictureMBS.ImageFormatRGBA, 1000000, f)
msgbox str(p.TotalSize/1024/1024)+" MB"
// this picture is 37 GB big!
```

Notes:

The size of this image is limited to available hard disc space.

The system will cache this data in memory to avoid writing it to disc. Using picture sizes bigger than physical memory can result into slow processing.

FilePath points to the location where the file is created

On Windows the FilePath can be nil in which space in the system swapfile is used.

BlockSize specifies how many bytes of memory should be used in application memory space. A typical value may be 100 mega bytes.

The Valid property is set to true on success.

See also:

- 2.1.1 Constructor(Buf as MemoryBlock, width as integer, height as integer, ImageFormat as integer, RowSize as integer) 46
- 2.1.1 Constructor(pic as picture, UseAlpha as boolean=false) 46
- 2.1.1 Constructor(width as integer, height as integer, ImageFormat as integer) 47

CopyGWorld as variant

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: No, Linux: No, . **Function:** Creates a GWorld with a copy of the picture data.

Example:

```
// get a PictureMBS object somewhere
dim pic as Picture = LogoMBS(500)
dim p as Picturembs = new PictureMBS(pic)

// make a gworld from it.
dim g as GWorldMBS = p.CopyGWorld

// for debugging show content
Backdrop = g.CopyPicture

// Export to TIFF with Quicktime Graphics Exporter
dim q as new QTGraphicsExporterMBS

q.OpenExporter("TIFF")
q.InputGWorldHandle = g.Handle
q.CompressionQuality = & h400
q.OutputFile = SpecialFolder.Desktop.Child("test.tif")

if q.Export>0 then
  MsgBox "OK"
end if
```

Notes: Works with 8 bit gray and with RGB data.

CopyMask as picture

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies the alpha channel into a mask picture.

Notes:

Be aware that PictureMBS objects can have more pixels than picture objects can store, so this will not always work.

Returns nil on any error (e.g. out of memory).

Works for all pictures with alpha channel.

See also:

- 2.1.1 CopyMask(x as integer, y as integer, w as integer, h as integer) as picture 50

CopyMask(x as integer, y as integer, w as integer, h as integer) as picture

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies the given area of the alpha channel into a mask picture.

Notes:

Be aware that PictureMBS objects can have more pixels than picture objects can store, so this will not always work.

Returns nil on any error (e.g. out of memory).

Works for all pictures with alpha channel.

See also:

- 2.1.1 CopyMask as picture 49

CopyPicture as picture

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies the RGB channels or the gray channel into a picture.

Example:

```
// get some picture
dim logo as Picture = LogoMBS(500)

// create PictureMBS
dim rgb as new PictureMBS(logo)

// Create a gray picture and copy RGB to gray
dim g as new PictureMBS(500, 500, PictureMBS.ImageFormatG)
call g.CopyPixels(rgb)

// Create CMYK and fill cyan channel with grayscale image
dim cmyk as new PictureMBS(500, 500, PictureMBS.ImageFormatCMYK)
call cmyk.MagentaChannel.CopyPixels(g)

// display it
Backdrop = cmyk.CopyPicture
```

Notes:

Be aware that PictureMBS objects can have more pixels than picture objects can store, so this will not always work.

Returns nil on any error (e.g. out of memory).

Works with Gray, RGB and CMYK pictures and supports alpha channel. For CMYK we have some simple conversion to RGB to give you a preview. For a real world application, use Color Conversion like our LCMS plugin.

See also:

- 2.1.1 CopyPicture(x as integer, y as integer, w as integer, h as integer) as picture 51

CopyPicture(x as integer, y as integer, w as integer, h as integer) as picture

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies the RGB channels or the gray channel in the given area into a picture.

Notes:

Be aware that PictureMBS objects can have more pixels than picture objects can store, so this will not always work.

Returns nil on any error (e.g. out of memory).

Works with Gray, RGB and CMYK pictures and supports alpha channel. For CMYK we have some simple conversion to RGB to give you a preview. For a real world application, use Color Conversion like our LCMS plugin.

See also:

- 2.1.1 CopyPicture as picture 50

CopyPictureWithMask as picture

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies the picture with mask.

Notes:

Be aware that PictureMBS objects can have more pixels than picture objects can store, so this will not always work.

Returns nil on any error (e.g. out of memory).

Works with Gray, RGB and CMYK pictures and supports alpha channel. For CMYK we have some simple

conversion to RGB to give you a preview. For a real world application, use Color Conversion like our LCMS plugin.

See also:

- 2.1.1 CopyPictureWithMask(x as integer, y as integer, w as integer, h as integer) as picture 52

CopyPictureWithMask(x as integer, y as integer, w as integer, h as integer) as picture

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies the picture with mask in the given area.

Notes:

Be aware that PictureMBS objects can have more pixels than picture objects can store, so this will not always work.

Returns nil on any error (e.g. out of memory).

Works with Gray, RGB and CMYK pictures and supports alpha channel. For CMYK we have some simple conversion to RGB to give you a preview. For a real world application, use Color Conversion like our LCMS plugin.

See also:

- 2.1.1 CopyPictureWithMask as picture 51

CopyPixels(source as PictureMBS) as boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies all pixels from the source picture to the current picture.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
// create new picture:
// can be any image format: ImageFormatRGB, ImageFormatG, ImageFormatBGR, etc.
dim d as new PictureMBS(l.Width, l.Height, PictureMBS.ImageFormatRGB)

if d.CopyPixels(p) then
Backdrop = d.CopyPicture
else
MsgBox "Failed."
end if
```

Notes:

This function is optimized for several image formats:

- Gray to Gray.
- RGB to Gray uses $R*0.3+G*0.59+B*0.11$.
- RGB to RGB.
- Gray to RGB fill red, green and blue with the same gray value.
- CMYK to CMYK
- CMYK to Gray, copies from black channel
- Gray to CMYK, copies to black channel

If an alpha channel exists in both images, it is copied.

See also:

- 2.1.1 CopyPixels(source as PictureMBS, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) as boolean 53
- 2.1.1 CopyPixels(source as PictureMBS, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) as boolean 54

CopyPixels(source as PictureMBS, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) as boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from the source picture to the current picture.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
dim d as new PictureMBS(700, 700, PictureMBS.ImageFormatRGB)

if d.CopyPixels(p,100,100,500,500) then
  Backdrop = d.CopyPicture
else
  MsgBox "Failed."
end if
```

Notes:

DestWidth and DestHeight specify how many pixels are copied.

DestX/DestY specify the destination position in the current picture.

This function is optimized for several image formats:

- Gray to Gray.
- RGB to Gray uses $R*0.3+G*0.59+B*0.11$.
- RGB to RGB.
- Gray to RGB fill red, green and blue with the same gray value.
- CMYK to Gray, copies from black channel
- Gray to CMYK, copies to black channel

If an alpha channel exists in both images, it is copied.

See also:

- 2.1.1 CopyPixels(source as PictureMBS) as boolean 52
- 2.1.1 CopyPixels(source as PictureMBS, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) as boolean 54

CopyPixels(source as PictureMBS, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) as boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Copies pixels from the source picture to the current picture.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
dim d as new PictureMBS(700, 700, PictureMBS.ImageFormatRGB)

if d.CopyPixels(p,100,100,500,500,0,0) then
  Backdrop = d.CopyPicture
else
  MsgBox "Failed."
end if
```

Notes:

SourceX/SourceY is the position in the source picture.
 DestWidth/DestHeight specify how many pixels are copied.
 DestX/DestY specify the destination position in the current picture.

This function is optimized for several image formats:

- Gray to Gray.

- RGB to Gray uses $R*0.3+G*0.59+B*0.11$.
- RGB to RGB.
- Gray to RGB fill red, green and blue with the same gray value.
- CMYK to Gray, copies from black channel
- Gray to CMYK, copies to black channel

If an alpha channel exists in both images, it is copied.

See also:

- 2.1.1 CopyPixels(source as PictureMBS) as boolean 52
- 2.1.1 CopyPixels(source as PictureMBS, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) as boolean 53

CreatePictureMBS(width as integer, height as integer, ImageFormat as integer) as PictureMBS

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a new PictureMBS object.

Notes:

Returns nil if no factory can create a valid picture.

First the Factory on the current ImageMBS object (self) is asked to create the picture.

Second the global Factory object is asked.

Third the normal PictureMBS constructor is used.

See also:

- 2.1.1 CreatePictureMBS(width as integer, height as integer, theImageFormat as integer) as PictureMBS 55

CreatePictureMBS(width as integer, height as integer, theImageFormat as integer) as PictureMBS

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a new PictureMBS object.

Notes:

Returns nil if no factory can create a valid picture.

First the global factory object is asked to create the picture.

Second the normal PictureMBS constructor is used.

See also:

- 2.1.1 CreatePictureMBS(width as integer, height as integer, ImageFormat as integer) as PictureMBS
55

CyanChannel as PictureMBS

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The cyan channel of a CMYK picture as a new PictureMBS object.

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

DataStringInFormat(ImageFormat as integer) as string

Plugin Version: 9.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The data of this picture as a string.

Notes:

Strings are limit to 2 GB, but the actual limit is certainly smaller.

You can get and set the image data with this method in the native format.

If you set the data, use a string with at least RowSize bytes.

If you query the data, you will get a copy of the data bytes in a string.

Returns "" on any error.

May raise OutOfBoundsException for invalid index.

(Read and Write computed property)

DiffuseFilter(dest as PictureMBS, level as integer) as PictureMBS

Plugin Version: 9.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies a diffuse filter to the image.

Notes:

if dest is nil, the picture factory is used to create a new picture.
 On success dest or the new picture is returned.
 If dest is not nil, it must match the size of the original picture.

Returns nil on any error.
 Level must be between 0 and min(width,height).

DitherFilter(dest as PictureMBS, matrix as integer, levels as integer) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies the dither filter to the picture.

Notes:

Use for the matrix parameter one of the Dither* constants.

if dest is nil, the picture factory is used to create a new picture.
 On success dest or the new picture is returned.
 If dest is not nil, it must match the size of the original picture.

Levels is a number between 2 and 256 and specifies how many color levels are in the final picture.

Returns nil on any error.

DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

This method applies the pixel values from the Red, Green and Blue channel of the picture with calculating in the mask of the picture.

This is the calculation:

$$\text{Pixel.Red} = (\text{Pixel.Red} * (255 - \text{Pixel.Mask}) + \text{PicturePixel.Red} * \text{PicturePixel.Mask}) / 255$$

```
Pixel.Green = (Pixel.Green * (255-Pixel.Mask) + PicturePixel.Green * PicturePixel.Mask) / 255
Pixel.Blue = (Pixel.Blue * (255-Pixel.Mask) + PicturePixel.Blue * PicturePixel.Mask) / 255
```

Works only if the PictureMBS has Red, Green and Blue channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, Dest-Width as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False) 58
- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, Invert-Mask as boolean=False) 59
- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, InvertMask as boolean=False) 59

DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, Dest-Width as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

This method applies the pixel values from the Red, Green and Blue channel of the picture with calculating in the mask of the picture.

This is the calculation:

```
Pixel.Red = (Pixel.Red * (255-Pixel.Mask) + PicturePixel.Red * PicturePixel.Mask) / 255
Pixel.Green = (Pixel.Green * (255-Pixel.Mask) + PicturePixel.Green * PicturePixel.Mask) / 255
Pixel.Blue = (Pixel.Blue * (255-Pixel.Mask) + PicturePixel.Blue * PicturePixel.Mask) / 255
```

Works only if the PictureMBS has Red, Green, Blue and Alpha channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, Dest-Width as integer, DestHeight as integer, InvertMask as boolean=False) 57
- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, Invert-Mask as boolean=False) 59
- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, InvertMask as boolean=False) 59

DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, InvertMask as boolean=False)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

This method applies the pixel values from the Red, Green and Blue channel of the picture with calculating in the mask of the picture.

This is the calculation:

$$\text{Pixel.Red} = (\text{Pixel.Red} * (255 - \text{Pixel.Mask}) + \text{PicturePixel.Red} * \text{PicturePixel.Mask}) / 255$$

$$\text{Pixel.Green} = (\text{Pixel.Green} * (255 - \text{Pixel.Mask}) + \text{PicturePixel.Green} * \text{PicturePixel.Mask}) / 255$$

$$\text{Pixel.Blue} = (\text{Pixel.Blue} * (255 - \text{Pixel.Mask}) + \text{PicturePixel.Blue} * \text{PicturePixel.Mask}) / 255$$

Works only if the PictureMBS has Red, Green, Blue and Alpha channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False) 57
- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False) 58
- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, InvertMask as boolean=False) 59

DrawMaskedPictureApplyMaskRGB(pic as picture, InvertMask as boolean=False)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

This method applies the pixel values from the Red, Green and Blue channel of the picture with calculating in the mask of the picture.

This is the calculation:

$$\text{Pixel.Red} = (\text{Pixel.Red} * (255 - \text{Pixel.Mask}) + \text{PicturePixel.Red} * \text{PicturePixel.Mask}) / 255$$

$$\text{Pixel.Green} = (\text{Pixel.Green} * (255 - \text{Pixel.Mask}) + \text{PicturePixel.Green} * \text{PicturePixel.Mask}) / 255$$

$\text{Pixel.Blue} = (\text{Pixel.Blue} * (255 - \text{Pixel.Mask}) + \text{PicturePixel.Blue} * \text{PicturePixel.Mask}) / 255$

Works only if the PictureMBS has Red, Green, Blue and Alpha channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False) 57
- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False) 58
- 2.1.1 DrawMaskedPictureApplyMaskRGB(pic as picture, DestX as integer, DestY as integer, InvertMask as boolean=False) 59

DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

This method copies the pixel values from the Red, Green and Blue channel of the picture and the pixel value of the picture's mask to the PictureMBS replacing old values.

This is the calculation:

Pixel.Red = PicturePixel.Red

Pixel.Green = PicturePixel.Green

Pixel.Blue = PicturePixel.Blue

Pixel.Alpha = PicturePixel.Mask

Works only if the PictureMBS has Red, Green, Blue and Alpha channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False) 61
- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, InvertMask as boolean=False) 61
- 2.1.1 DrawMaskedPictureRGB(pic as picture, InvertMask as boolean=False) 62

DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

This method copies the pixel values from the Red, Green and Blue channel of the picture and the pixel value of the picture's mask to the PictureMBS replacing old values.

This is the calculation:

```
Pixel.Red = PicturePixel.Red
Pixel.Green = PicturePixel.Green
Pixel.Blue = PicturePixel.Blue
Pixel.Alpha = PicturePixel.Mask
```

Works only if the PictureMBS has Red, Green and Blue channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False) 60
- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, InvertMask as boolean=False) 61
- 2.1.1 DrawMaskedPictureRGB(pic as picture, InvertMask as boolean=False) 62

DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, InvertMask as boolean=False)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

This method copies the pixel values from the Red, Green and Blue channel of the picture and the pixel value of the picture's mask to the PictureMBS replacing old values.

This is the calculation:

```
Pixel.Red = PicturePixel.Red
Pixel.Green = PicturePixel.Green
```

```
Pixel.Blue = PicturePixel.Blue
Pixel.Alpha = PicturePixel.Mask
```

Works only if the PictureMBS has Red, Green, Blue and Alpha channels.
If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.
See also:

- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False) 60
- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False) 61
- 2.1.1 DrawMaskedPictureRGB(pic as picture, InvertMask as boolean=False) 62

DrawMaskedPictureRGB(pic as picture, InvertMask as boolean=False)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

This method copies the pixel values from the Red, Green and Blue channel of the picture and the pixel value of the picture's mask to the PictureMBS replacing old values.

This is the calculation:

```
Pixel.Red = PicturePixel.Red
Pixel.Green = PicturePixel.Green
Pixel.Blue = PicturePixel.Blue
Pixel.Alpha = PicturePixel.Mask
```

Works only if the PictureMBS has Red, Green, Blue and Alpha channels.
If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.
See also:

- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, InvertMask as boolean=False) 60
- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer, InvertMask as boolean=False) 61
- 2.1.1 DrawMaskedPictureRGB(pic as picture, DestX as integer, DestY as integer, InvertMask as boolean=False) 61

DrawPictureBlueToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws the blue channel of a picture object into the gray channel of this picture.

Notes:

If you want to copy the blue channel of the picture into the blue channel of the PictureMBS, then first get a PictureMBS object for the blue channel and use this method on this object.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawPictureBlueToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) 63

DrawPictureBlueToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws the blue channel of a picture object into the gray channel of this picture.

Notes:

If you want to copy the blue channel of the picture into the blue channel of the PictureMBS, then first get a PictureMBS object for the blue channel and use this method on this object.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawPictureBlueToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) 63

DrawPictureGreenToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws the green channel of a picture object into the gray channel of this picture.

Notes:

If you want to copy the green channel of the picture into the green channel of the PictureMBS, then first get a PictureMBS object for the green channel and use this method on this object.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawPictureGreenToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) 64

DrawPictureGreenToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws the green channel of a picture object into the gray channel of this picture.

Notes:

If you want to copy the green channel of the picture into the green channel of the PictureMBS, then first get a PictureMBS object for the green channel and use this method on this object.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawPictureGreenToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) 63

DrawPictureRedToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws the red channel of a picture object into the gray channel of this picture.

Notes:

If you want to copy the red channel of the picture into the red channel of the PictureMBS, then first get a PictureMBS object for the red channel and use this method on this object.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawPictureRedToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) 64

DrawPictureRedToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws the red channel of a picture object into the gray channel of this picture.

Notes:

If you want to copy the red channel of the picture into the red channel of the PictureMBS, then first get a PictureMBS object for the red channel and use this method on this object.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

See also:

- 2.1.1 DrawPictureRedToGrayChannel(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) 64

DrawPictureRGB(pic as picture)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

Works only if the PictureMBS has Red, Green and Blue channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

This is the calculation:

Pixel.Red = PicturePixel.Red

Pixel.Green = PicturePixel.Green

Pixel.Blue = PicturePixel.Blue

This method does ignore a mask in the given picture and does not change set the alpha channel.

See also:

- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer) 65
- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) 66
- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) 67

DrawPictureRGB(pic as picture, DestX as integer, DestY as integer)

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

Works only if the PictureMBS has Red, Green and Blue channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

This is the calculation:

```
Pixel.Red = PicturePixel.Red
Pixel.Green = PicturePixel.Green
Pixel.Blue = PicturePixel.Blue
```

This method does ignore a mask in the given picture and does not change set the alpha channel.

See also:

- 2.1.1 DrawPictureRGB(pic as picture) 65
- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) 66
- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) 67

DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

Works only if the PictureMBS has Red, Green and Blue channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

This is the calculation:

```
Pixel.Red = PicturePixel.Red
Pixel.Green = PicturePixel.Green
Pixel.Blue = PicturePixel.Blue
```

This method does ignore a mask in the given picture and does not change set the alpha channel.

See also:

- 2.1.1 DrawPictureRGB(pic as picture) 65
- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer) 65

- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer) 67

DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer, SourceX as integer, SourceY as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Draws a picture into this PictureMBS object.

Notes:

Works only if the PictureMBS has Red, Green and Blue channels.

If you want to copy Pixels from a PictureMBS to a PictureMBS, use CopyPixels.

This is the calculation:

Pixel.Red = PicturePixel.Red

Pixel.Green = PicturePixel.Green

Pixel.Blue = PicturePixel.Blue

This method does ignore a mask in the given picture and does not change set the alpha channel.

See also:

- 2.1.1 DrawPictureRGB(pic as picture) 65
- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer) 65
- 2.1.1 DrawPictureRGB(pic as picture, DestX as integer, DestY as integer, DestWidth as integer, DestHeight as integer) 66

EngraveFilter(dest as PictureMBS, level as integer) as PictureMBS

Plugin Version: 9.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies an engrave filter to the image.

Notes:

if dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.

If dest is not nil, it must match the size of the original picture.

Level must be between 0 and min(width,height).

Returns nil on any error.

FillRect(value as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRect(200)
window1.Backdrop = p.CopyPicture
```

Notes:

All channels are filled with the given value.

The range of value is 0 to 255.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 FillRect(x as integer, y as integer, width as integer, height as integer, value as integer) 68

FillRect(x as integer, y as integer, width as integer, height as integer, value as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the given area of the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRect(10, 10, 20, 20, 200)
window1.Backdrop = p.CopyPicture
```

Notes:

All channels are filled with the given value.

The range of value is 0 to 255.

Works with Gray, RGB and CMYK pictures and supports alpha channel.
See also:

- 2.1.1 FillRect(value as integer)

68

FillRectRandom

Plugin Version: 10.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the pixels with random values.

Example:

```
dim p as new PictureMBS(1000,1000, PictureMBS.imageFormatRGB)
```

```
p.FillRectRandom
```

Notes: Works with Gray, RGB and CMYK pictures and supports alpha channel.
See also:

- 2.1.1 FillRectRandom(x as integer, y as integer, width as integer, height as integer)

69

FillRectRandom(x as integer, y as integer, width as integer, height as integer)

Plugin Version: 10.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the pixels with random values.

Example:

```
dim p as new PictureMBS(1000,1000, PictureMBS.imageFormatRGB)
```

```
p.FillRectRandom(0,0,100,100)
```

```
Backdrop = p.CopyPicture
```

Notes: Works with Gray, RGB and CMYK pictures and supports alpha channel.
See also:

- 2.1.1 FillRectRandom

69

FillRectRGB(FillColor as color)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRectRGB(& cFF0000)
window1.Backdrop = p.CopyPicture
```

Notes: Works only if the picture has RGB channels.

See also:

- 2.1.1 FillRectRGB(FillColor as color, alpha as integer) 70
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer) 71
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer) 72
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer) 74
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer) 75

FillRectRGB(FillColor as color, alpha as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRectRGB(& cFF0000, 0)
window1.Backdrop = p.CopyPicture
```

Notes:

Works only if the picture has RGB channels.
 Alpha is ignored if the picture does not have an alpha channel.
 The range of alpha is 0 to 255.
 See also:

- 2.1.1 FillRectRGB(FillColor as color) 70
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer) 71
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer) 72
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer) 74
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer) 75

FillRectRGB(red as integer, green as integer, blue as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRectRGB(255, 0, 0)
window1.Backdrop = p.CopyPicture
```

Notes:

Works only if the picture has RGB channels.
 The ranges of red, green and blue are 0 to 255.
 See also:

- 2.1.1 FillRectRGB(FillColor as color) 70
- 2.1.1 FillRectRGB(FillColor as color, alpha as integer) 70

- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer) 72
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer) 74
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer) 75

FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRectRGB(255, 0, 0, 0)
window1.Backdrop = p.CopyPicture
```

Notes:

Works only if the picture has RGB channels.
Alpha is ignored if the picture does not have an alpha channel.
The ranges of alpha, red, green and blue are 0 to 255.
See also:

- 2.1.1 FillRectRGB(FillColor as color) 70
- 2.1.1 FillRectRGB(FillColor as color, alpha as integer) 70
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer) 71
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer) 74
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer) 75

FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the given area of the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRectRGB(10,10,20,20,& cFF0000)
window1.Backdrop = p.CopyPicture
```

Notes: Works only if the picture has RGB channels.

See also:

- 2.1.1 FillRectRGB(FillColor as color) 70
- 2.1.1 FillRectRGB(FillColor as color, alpha as integer) 70
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer) 71
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer) 72
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer) 74
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer) 75

FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the given area of the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRectRGB(10,10,20,20,& cFF0000, 0)
window1.Backdrop = p.CopyPicture
```

Notes:

Works only if the picture has RGB channels.
 Alpha is ignored if the picture does not have an alpha channel.
 The range of alpha is 0 to 255.
 See also:

- 2.1.1 FillRectRGB(FillColor as color) 70
- 2.1.1 FillRectRGB(FillColor as color, alpha as integer) 70
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer) 71
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer) 72
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer) 74
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer) 75

FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the given area of the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRectRGB(10,10,20,20, 255, 0, 0)
window1.Backdrop = p.CopyPicture
```

Notes:

Works only if the picture has RGB channels.
 The ranges of red, green and blue are 0 to 255.
 See also:

- 2.1.1 FillRectRGB(FillColor as color) 70
- 2.1.1 FillRectRGB(FillColor as color, alpha as integer) 70

- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer) 71
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer) 72
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer) 75

FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer, alpha as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Fills the given area of the picture with the given color.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.FillRectRGB(10, 10, 20, 20, 255, 0, 0, 0)
window1.Backdrop = p.CopyPicture
```

Notes:

Works only if the picture has RGB channels.
Alpha is ignored if the picture does not have an alpha channel.
See also:

- 2.1.1 FillRectRGB(FillColor as color) 70
- 2.1.1 FillRectRGB(FillColor as color, alpha as integer) 70
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer) 71
- 2.1.1 FillRectRGB(red as integer, green as integer, blue as integer, alpha as integer) 72
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, FillColor as color, alpha as integer) 73
- 2.1.1 FillRectRGB(x as integer, y as integer, width as integer, height as integer, red as integer, green as integer, blue as integer) 74

GainFilter(dest as PictureMBS, gain as double, bias as double) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies the gain filter to the picture.

Notes:

if dest is nil, the picture factory is used to create a new picture.
On success dest or the new picture is returned.
If dest is not nil, it must match the size of the original picture.

Returns nil on any error.

GammaFilter(dest as PictureMBS, gamma as double) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Changes the gamma value of the picture.

Notes:

If dest is nil, the picture factory is used to create a new picture.
On success dest or the new picture is returned.
If dest is not nil, it must match the size of the original picture.

Returns nil on any error.

See also:

- 2.1.1 GammaFilter(dest as PictureMBS, gamma as double, alphaGamma as double) as PictureMBS
76
- 2.1.1 GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double) as PictureMBS
77
- 2.1.1 GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double, alphaGamma as double) as PictureMBS
78

GammaFilter(dest as PictureMBS, gamma as double, alphaGamma as double) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Changes the gamma value of the picture.

Notes:

If dest is nil, the picture factory is used to create a new picture.
 On success dest or the new picture is returned.
 If dest is not nil, it must match the size of the original picture.

If the picture has no alpha channel, the alpha parameter is ignored.

Returns nil on any error.
 See also:

- 2.1.1 GammaFilter(dest as PictureMBS, gamma as double) as PictureMBS 76
- 2.1.1 GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double) as PictureMBS 77
- 2.1.1 GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double, alphaGamma as double) as PictureMBS 78

GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Changes the gamma value of the picture.

Notes:

If dest is nil, the picture factory is used to create a new picture.
 On success dest or the new picture is returned.
 If dest is not nil, it must match the size of the original picture.

For grayscale pictures the gray color is calculated from red, green and blue value.

Returns nil on any error.
 See also:

- 2.1.1 GammaFilter(dest as PictureMBS, gamma as double) as PictureMBS 76
- 2.1.1 GammaFilter(dest as PictureMBS, gamma as double, alphaGamma as double) as PictureMBS 76
- 2.1.1 GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double, alphaGamma as double) as PictureMBS 78

GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double, alphaGamma as double) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Changes the gamma value of the picture.

Notes:

If dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.

If dest is not nil, it must match the size of the original picture.

For grayscale pictures the gray color is calculated from red, green and blue value.

If the picture has no alpha channel, the alpha parameter is ignored.

Returns nil on any error.

See also:

- 2.1.1 GammaFilter(dest as PictureMBS, gamma as double) as PictureMBS 76
- 2.1.1 GammaFilter(dest as PictureMBS, gamma as double, alphaGamma as double) as PictureMBS 76
- 2.1.1 GammaFilter(dest as PictureMBS, redGamma as double, greenGamma as double, blueGamma as double) as PictureMBS 77

GrayChannel as PictureMBS

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The gray channel as a new PictureMBS object.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
dim r as PictureMBS = p.GrayChannel
r.fillrect(100) // fill only gray channel
```

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

GreenChannel as PictureMBS

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The green channel as a new PictureMBS object.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
dim r as PictureMBS = p.GreenChannel
r.fillrect(100) // fill only green channel
```

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

HMirror

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Mirrors the image content horizontally (flip).

Example:

```
// get some picture
dim l as Picture = LogoMBS(500)
// create new image
dim p as new PictureMBS(l)
// mirror
p.HMirror
// show in window
window1.Backdrop = p.CopyPicture
```

Notes: Works with Gray, RGB and CMYK pictures and supports alpha channel.

Invert

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Inverts the image data.

Notes: Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 Invert(x as integer, y as integer, w as integer, h as integer) 80

Invert(x as integer, y as integer, w as integer, h as integer)

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Inverts the image data in the given area.

Notes: Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 Invert 80

MagentaChannel as PictureMBS

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The magenta channel of a CMYK picture as a new PictureMBS object.

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

MirroredView as PictureMBS

Plugin Version: 10.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a new PictureMBS object which draws into the existing one, but has all rows vertically mirrored.

Notes: So if the new picture draws into the first row, the change will be in the last row of the original picture.

NeonFilter(dest as PictureMBS) as PictureMBS

Plugin Version: 9.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies a neon filter to the image.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p = p.NeonFilter(nil)
window1.Backdrop = p.CopyPicture
```

Notes:

if dest is nil, the picture factory is used to create a new picture.
On success dest or the new picture is returned.
If dest is not nil, it must match the size of the original picture.

Returns nil on any error.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

OilFilter(dest as PictureMBS, levels as integer, range as integer) as PictureMBS

Plugin Version: 9.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies a oil filter to the image.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p = p.OilFilter(nil,5,5)
window1.Backdrop = p.CopyPicture
```

Notes:

if dest is nil, the picture factory is used to create a new picture.
On success dest or the new picture is returned.
If dest is not nil, it must match the size of the original picture.

Levels must be between 0 and 256.
 Range must be between 0 and min(width,height).

Returns nil on any error.
 Works with Gray, RGB and CMYK pictures and supports alpha channel.

RawRow(index as integer) as memoryblock

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns a memoryblock with the data of this row.

Example:

```
// create new image
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
// copy row
dim m as MemoryBlock = p.RawRow(10)
// modify directly
m.FillBytesMBS(10,100,200)
// show in window
window1.Backdrop = p.CopyPicture
```

Notes:

This memoryblock is pointing to the original data, so any modification is applied to the picture.
 Returns nil on any error.
 May raise OutOfBoundsException for invalid index.

For pictures using virtual memory, this memoryblock can become invalid for the next call to any PictureMBS method!

RedChannel as PictureMBS

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The red channel as a new PictureMBS object.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
dim r as PictureMBS = p.RedChannel
r.fillrect(100) // fill only red channel
```

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

RGBChannels as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The RGB channels as a new PictureMBS object.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGBA)
dim r as PictureMBS = p.RGBChannels
r.fillrect(100) // fill only color channels
```

Notes:

Returns nil if the image is not a RGB picture.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the RGB pixels directly without modifying an alpha channel

The resulting PictureMBS object is a RGB picture.

Rotate(angle as double, Red as integer = 0, Green as integer = 0, Blue as integer = 0, Alpha as integer = 0, Gray as integer = 0, Cyan as integer = 0, Magenta as integer = 0, Yellow as integer = 0, Black as integer = 0) as PictureMBS

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Rotates the picture by the given degree.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p = p.Rotate(30, 255, 255, 255, 255, 255)
window1.Backdrop = p.CopyPicture
```

Notes:

With Red, Blue, Green, Alpha and Gray specify the color of the fill color.

If dest is nil, the PictureFactoryMBS object (local on self or global) is used to create the new picture. Works with Gray, RGB and CMYK pictures and supports alpha channel.

Rotate180

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Rotates the picture by 180 degree.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p.Rotate180
window1.Backdrop = p.CopyPicture
```

Notes:

Same as HMirror and VMirror together.

There are two Rotate180 methods. One makes a copy and one not. This one does not make a copy.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 Rotate180(dest as PictureMBS=nil) as PictureMBS

85

Rotate180(dest as PictureMBS=nil) as PictureMBS

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Rotates the picture by 180 degree.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p = p.Rotate180
window1.Backdrop = p.CopyPicture
```

Notes:

If dest is nil, the PictureFactoryMBS object (local on self or global) is used to create the new picture.

Same as HMirror and VMirror together.

There are two Rotate180 methods. One makes a copy and one not. This one does make a copy.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

See also:

- 2.1.1 Rotate180

84

Rotate270(dest as PictureMBS=nil) as PictureMBS

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Rotates the picture by 270 degree.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p = p.Rotate270
window1.Backdrop = p.CopyPicture
```

Notes:

If dest is nil, the PictureFactoryMBS object (local on self or global) is used to create the new picture. Works with Gray, RGB and CMYK pictures and supports alpha channel.

Rotate270slow(dest as PictureMBS=nil) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Rotates the picture by 270 degree.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p = p.Rotate270slow
window1.Backdrop = p.CopyPicture
```

Notes:

If dest is nil, the PictureFactoryMBS object (local on self or global) is used to create the new picture. Works with Gray, RGB and CMYK pictures and supports alpha channel.

Rotate90(dest as PictureMBS=nil) as PictureMBS

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Rotates the picture by 90 degree.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p = p.Rotate90
window1.Backdrop = p.CopyPicture
```

Notes:

If dest is nil, the PictureFactoryMBS object (local on self or global) is used to create the new picture. Works with Gray, RGB and CMYK pictures and supports alpha channel.

Rotate90slow(dest as PictureMBS=nil) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Rotates the picture by 90 degree.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
p = p.Rotate90slow
window1.Backdrop = p.CopyPicture
```

Notes:

If dest is nil, the PictureFactoryMBS object (local on self or global) is used to create the new picture. Works with Gray, RGB and CMYK pictures and supports alpha channel.

Row(index as integer) as memoryblock

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A memoryblock with the data of this row.

Example:

```
// create new image
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
// copy row
dim m as MemoryBlock = p.Row(10)
// modify
m.FillBytesMBS(10,100,200)
// copy back
p.row(10)=m
// show in window
```

```
window1.Backdrop = p.CopyPicture
```

Notes:

You can get and set a row with this method in the native format.
 If you set the row, use a memoryblock with at least RowSize bytes.
 If you query the row, you will get a copy of the row bytes in a new memoryblock.
 Returns "" on any error.
 May raise OutOfBoundsException for invalid index.
 (Read and Write computed property)

RowInFormat(index as integer, ImageFormat as integer) as memoryblock

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A memoryblock with the data of this row in the format you request.

Example:

```
// create new image
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
// copy row
dim m as MemoryBlock = p.RowInFormat(10, p.ImageFormatRofRGB)
// modify
m.FillBytesMBS(10,80,200)
// copy back
p.RowInFormat(10, p.ImageFormatRofRGB)=m
// show in window
window1.Backdrop = p.CopyPicture
```

Notes:

You can get and set a row with this method in the given format.
 If you set the row, use a memoryblock with at least Width*PixelSize bytes. PixelSize is the format dependend size in bytes for one pixel.
 If you query the row, you will get a copy of the row bytes in a new memoryblock.
 Returns nil on any error.
 May raise OutOfBoundsException for invalid index.
 (Read and Write computed property)
 See also:

- 2.1.1 RowInFormat(index as integer, ImageFormat as integer, InvertAlpha as boolean) as memoryblock
89

RowInFormat(index as integer, ImageFormat as integer, InvertAlpha as boolean) as memory-block

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A memoryblock with the data of this row in the format you request.

Example:

```
// create new image
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
// copy row
dim m as MemoryBlock = p.RowInFormat(10, p.ImageFormatRGB, true)
// modify
m.FillBytesMBS(10,80,200)
// copy back
p.RowInFormat(10, p.ImageFormatRGB, true)=m
// show in window
window1.Backdrop = p.CopyPicture
```

Notes:

You can get and set a row with this method in the given format.

If you set the row, use a memoryblock with at least Width*PixelSize bytes. PixelSize is the format dependend size in bytes for one pixel.

If you query the row, you will get a copy of the row bytes in a new memoryblock.

Returns nil on any error.

May raise OutOfBoundsException for invalid index.

If InvertAlpha is true, the alpha values are inverted by using $A=255-A$.

(Read and Write computed property)

See also:

- 2.1.1 RowInFormat(index as integer, ImageFormat as integer) as memoryblock

RowStringInFormat(index as integer, ImageFormat as integer) as string

Plugin Version: 9.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The row as a string.

Notes:

You can get and set a row with this method in the native format.
 If you set the row, use a memoryblock with at least RowSize bytes.
 If you query the row, you will get a copy of the row bytes in a string.
 Returns nil on any error.
 May raise OutOfBoundsException for invalid index.
 (Read and Write computed property)

Scale(source as PictureMBS, temp as PictureMBS, mode as integer, width as integer, height as integer) as boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Scales the picture to the given size.

Notes:

The final image is stored in the PictureMBS object you call this method on.
 On low memory this function can fail or the image may look bad. (e.g. all black)

The size of the temporary picture must have the size of the destination width and the source height. Use ImageFormatScaling when you create the temp image to give it the correct size.

For scaling with the same size as the picture already has, the scaling is still performed.

Returns true on success and false on any error. (e.g. width=0)

Use the constants for the mod:

ScaleTriangle	triangle
ScaleBox	box, nearest neighbor
ScaleLanczos3	lanczos 3
ScaleLanczos8	lanczos 8
ScaleMitchell	mitchell
ScalePoly3	poly 3
ScaleCubic	cubic

This function is optimized for several image formats:

- Gray to Gray.
- RGB to Gray uses $R*0.3+G*0.59+B*0.11$.
- RGB to RGB.

- Gray to RGB fill red, green and blue with the same gray value.
If an alpha channel exists in both images, it is copied.

ScaleFast(source as PictureMBS, width as integer, height as integer) as boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Scales the picture to the new size fast.

Notes:

The final image is stored in the PictureMBS object you call this method on.
Returns true on success and false on failure.

This is a low quality algorithm, but it is fast.

This function is optimized for several image formats:

- Gray to Gray.
 - RGB to Gray uses $R*0.3+G*0.59+B*0.11$.
 - RGB to RGB.
 - Gray to RGB fill red, green and blue with the same gray value.
- If an alpha channel exists in both images, it is copied.

ScaleMT(threads as integer, source as PictureMBS, temp as PictureMBS, mode as integer, width as integer, height as integer) as boolean

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The multithreaded variant of Scale function.

Notes:

Same as Scale, but with additional multithreading.
Must be called inside a Real Studio thread so time yields to main thread and you can keep the GUI running.
Work is split into several threads for greater speed.

Threads parameter specifies how many threads you want to use:

A negative value disables threading, zero will use one thread for each CPU core and a positive number specifies the thread count.

If one of the pictures used has `IsMapping = true`, the plugin calls `Scale()` function.

SolarizeFilter(dest as PictureMBS) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies the solarize filter to the picture.

Example:

```
// get some picture
dim l as Picture = LogoMBS(500)
// create new image
dim p as new PictureMBS(l)
// add filter
p = p.SolarizeFilter(nil)
// show in window
window1.Backdrop = p.CopyPicture
```

Notes:

if dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.

If dest is not nil, it must match the size of the original picture.

Returns nil on any error.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

StampFilter(dest as PictureMBS, radius as double, threshold as double, softness as double, Black as Color, White as Color) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies the stamp filter to the picture.

Notes:

if dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.

If dest is not nil, it must match the size of the original picture.

Returns nil on any error.

Works with Gray and RGB pictures and supports alpha channel.

TransferFilter(dest as PictureMBS, gray() as integer) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Transfers a picture to another picture by looking up each pixel value in the given array.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)

dim gray(256) as integer

for i as integer = 0 to 255
gray(i)=255-i // invert
next

// inverts the picture
dim d as PictureMBS = p.TransferFilter(nil,gray)

Backdrop = d.CopyPicture
```

Notes:

if dest is nil, the picture factory is used to create a new picture.
On success dest or the new picture is returned.
If dest is not nil, it must match the size of the original picture.

The array for gray must have 256 entries starting with index 0.
For RGB pictures the gray array is used for all three channels.

Returns nil on any error.
See also:

- 2.1.1 TransferFilter(dest as PictureMBS, gray() as integer, alpha() as integer) as PictureMBS 94
- 2.1.1 TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer) as PictureMBS 94
- 2.1.1 TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer, alpha())

as integer) as PictureMBS

95

TransferFilter(dest as PictureMBS, gray() as integer, alpha() as integer) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Transfers a picture to another picture by looking up each pixel value in the given arrays.

Notes:

if dest is nil, the picture factory is used to create a new picture.

On success dest or the new picture is returned.

If dest is not nil, it must match the size of the original picture.

The arrays for gray and alpha must have 256 entries starting with index 0.

For RGB pictures the gray array is used for all three channels.

If the picture has no alpha channel, the alpha parameter is ignored.

Returns nil on any error.

See also:

- 2.1.1 TransferFilter(dest as PictureMBS, gray() as integer) as PictureMBS 93
- 2.1.1 TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer) as PictureMBS 94
- 2.1.1 TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer, alpha() as integer) as PictureMBS 95

TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Transfers a picture to another picture by looking up each pixel value in the given arrays.

Example:

```
dim l as Picture = LogoMBS(500)
```

```
dim p as new PictureMBS(l)
```

```
dim red(256) as integer
```

```
dim green(256) as integer
```

```
dim blue(256) as integer
```

```

for i as integer = 0 to 255
red(i)=i
green(i)=i
blue(i)=255-i // invert blue
next

dim d as PictureMBS = p.TransferFilter(nil,red,green,blue)

Backdrop = d.CopyPicture

```

Notes:

if dest is nil, the picture factory is used to create a new picture.
On success dest or the new picture is returned.
If dest is not nil, it must match the size of the original picture.

The arrays for red, green and blue must have 256 entries starting with index 0.
For grayscale pictures the green array is used for the gray channel.

Returns nil on any error.
See also:

- 2.1.1 TransferFilter(dest as PictureMBS, gray() as integer) as PictureMBS 93
- 2.1.1 TransferFilter(dest as PictureMBS, gray() as integer, alpha() as integer) as PictureMBS 94
- 2.1.1 TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer, alpha() as integer) as PictureMBS 95

TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer, alpha() as integer) as PictureMBS

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Transfers a picture to another picture by looking up each pixel value in the given arrays.

Notes:

if dest is nil, the picture factory is used to create a new picture.
On success dest or the new picture is returned.
If dest is not nil, it must match the size of the original picture.

The arrays for red, green, blue and alpha must have 256 entries starting with index 0.
 For grayscale pictures the green array is used for the gray channel.
 If the picture has no alpha channel, the alpha parameter is ignored.

Returns nil on any error.
 See also:

- 2.1.1 TransferFilter(dest as PictureMBS, gray() as integer) as PictureMBS 93
- 2.1.1 TransferFilter(dest as PictureMBS, gray() as integer, alpha() as integer) as PictureMBS 94
- 2.1.1 TransferFilter(dest as PictureMBS, red() as integer, green() as integer, blue() as integer) as PictureMBS 94

UnsharpFilter(origpixels as PictureMBS, Amount as double, Threshold as integer) as boolean

Plugin Version: 9.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Runs the unsharp filter.

Notes:

You may want to run the BoxBlur filter first before using the unsharp filter.

if dest is nil, the picture factory is used to create a new picture.
 On success dest or the new picture is returned.
 If dest is not nil, it must match the size of the original picture.

Returns nil on any error.

Works with Gray, RGB and CMYK pictures and supports alpha channel.

VMirror

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Mirrors the image content vertically.

Example:

```
// get some picture
dim l as Picture = LogoMBS(500)
```

```
// create new image
dim p as new PictureMBS(1)
// mirror
p.VMirror
// show in window
window1.Backdrop = p.CopyPicture
```

Notes: Works with Gray, RGB and CMYK pictures and supports alpha channel.

YellowChannel as PictureMBS

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The yellow channel of a CMYK picture as a new PictureMBS object.

Notes:

Returns nil if this channel does not exist.

No copy is made of the actual pixel data. Modifying the channel picture will modify the original picture.

Use this function to access the pixels of the channel directly.

The resulting PictureMBS object is a grayscale picture.

2.1.2 Properties

AlphaOffset as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the alpha channel.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGA)
MsgBox str(p.AlphaOffset)
```

Notes: (Read only property)

BlackOffset as Integer

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the black channel.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatCMYK)
MsgBox str(p.BlackOffset)
```

Notes: (Read only property)

BlueOffset as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the blue channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.BlueOffset)
```

Notes: (Read only property)

ChannelCount as Integer

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The number of channels in this picture.

Notes:

1 for gray, 2 for gray+alpha, 3 for RGB, 4 for RGB+alpha or CMYK and 5 for CMYK+alpha.
(Read only property)

CyanOffset as Integer

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the cyan channel.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatCMYK)
MsgBox str(p.CyanOffset)
```

Notes: (Read only property)

DebugPicture as Picture

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The picture content to view in the debugger.

Notes:

If DebugPictureEnabled is set to true in our code you can use the DebugPicture property to watch the picture content in the debugger. For speed reasons the size of the debug picture is limited to 512 by 512 pixels. (that could be increased)

(Read only property)

DebugPictureEnabled as Boolean

Plugin Version: 9.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether you want to use the DebugPicture property.

Notes:

If DebugPictureEnabled is set to true in our code you can use the DebugPicture property to watch the picture content in the debugger. For speed reasons the size of the debug picture is limited to 512 by 512 pixels. (that could be increased)

(Read and Write property)

Factory as PictureFactoryMBS

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The local factory to be used for pictures created in this picture.

Notes:

If one of the functions in this PictureMBS instance needs a new PictureMBS object, this factory is asked first.

(Read and Write property)

GrayOffset as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the gray channel.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatAG)
MsgBox str(p.GrayOffset)
```

Notes: (Read only property)

GreenOffset as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the green channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.GreenOffset)
```

Notes: (Read only property)

HasAlpha as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has an alpha channel.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGA)
MsgBox str(p.HasAlpha)
```

Notes: (Read only property)

HasBlack as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has a blue channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.HasBlack)
```

Notes: (Read only property)

HasBlue as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has a blue channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.HasBlue)
```

Notes: (Read only property)

HasCyan as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has a blue channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.HasCyan)
```

Notes: (Read only property)

HasGray as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has a gray channel.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGA)
MsgBox str(p.HasGray)
```

Notes: (Read only property)

HasGreen as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has a green channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.HasGreen)
```

Notes: (Read only property)

HasMagenta as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has a blue channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.HasMagenta)
```

Notes: (Read only property)

HasRed as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has a channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.HasRed)
```

Notes: (Read only property)

HasYellow as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture has a blue channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.HasYellow)
```

Notes: (Read only property)

Height as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The height of the picture in pixels.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.Height)
```

Notes: (Read only property)

ImageFormat as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The image format of this picture object.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.ImageFormat)
```

Notes:

See the ImageFormat* constants.
(Read only property)

ImageFormatString as String

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The format of this picture as a string.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox p.ImageFormatString
```

Notes:

Returns for example "RGB" for ImageFormatRGB.
(Read only property)

IsCMYK as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture is a CMYK picture.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatCMYK)
MsgBox str(p.IsCMYK)
```

Notes:

HasCyan, HasMagenta, HasYellow and HasBlack are true if IsRGB is true.
(Read only property)

IsGray as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether this picture is a grayscale picture.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGA)
MsgBox str(p.IsGray)
```

Notes:

HasGray is true if IsGray is true.
(Read only property)

IsMapping as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether this picture uses virtual memory.

Notes:

If IsMapping is true you should not use the Memory property or the Clone function.
(Read only property)

IsRGB as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the picture is a RGB picture.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.IsRGB)
```

Notes:

HasRed, HasBlue and HasGreen are true if IsRGB is true.
(Read only property)

MagentaOffset as Integer

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the magenta channel.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatCMYK)
MsgBox str(p.MagentaOffset)
```

Notes: (Read only property)

MappingBlockSize as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The block size for a picture using virtual memory.

Notes: (Read and Write property)

Memory as Memoryblock

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns a memoryblock without size pointing to the current pixel buffer.

Notes:

Use only if IsMapping is false.
(Read only property)

MemoryTarget as Memoryblock

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** If this picture stores its pixels in a memoryblock, you can access the memory block using this property.

Notes: (Read only property)

Parent as PictureMBS

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The parent PictureMBS object.

Notes:

One PictureMBS can reference the pixels of another PictureMBS. The parent is referenced in this property so it is not released.

(Read only property)

PixelSize as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The size of a pixel in bytes.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.PixelSize)
```

Notes:

For example:

1 for Gray

2 for Gray with Alpha

3 for RGB

4 for RGB with Alpha

(Read only property)

RedOffset as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the red channel.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.RedOffset)
```

Notes: (Read only property)

RowOffset as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal row offset.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
dim q as PictureMBS = p.ClipImage(10,10,80,80)
```

```
MsgBox str(q.width)+" x "+str(q.height)+" with row offset: "+str(q.RowOffset)
```

Notes:

Only used with clipping images.
(Read only property)

RowSize as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The size of one row in bytes.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.RowSize)
```

Notes:

Additional bytes may be needed per row for better alignment of the data.
Also using virtual memory functions requires alignment.

(Read only property)

Target as Picture

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The target picture.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
```

```
window1.Backdrop = p.Target
```

Notes:

if this PictureMBS references the pixels of a REALbasic picture, this property keeps a reference to this target picture.

(Read only property)

TotalSize as Int64

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The total size of this picture in bytes.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
```

```
MsgBox str(p.TotalSize)
```

Notes:

The result is Height*RowSize.

(Read only property)

Valid as Boolean

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether this instance is a valid picture.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.Valid)
```

Notes:

Valid is false if the constructor failed to create a picture.

(Read only property)

Width as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The width of the picture in pixels.

Example:

```
dim l as Picture = LogoMBS(500)
dim p as new PictureMBS(l)
MsgBox str(p.Width)
```

Notes: (Read only property)

YellowOffset as Integer

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal offset for pixels in the yellow channel.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatCMYK)
MsgBox str(p.YellowOffset)
```

Notes: (Read only property)

YieldTicks as Integer

Plugin Version: 8.7 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** How much time is given back to REALbasic for other ticks.

Example:

```
dim p as PictureMBS // your picture  
  
p.YieldTicks=6 // only use 1/10th of a second
```

Notes:

If value is greater than zero, the application will yield to another RB thread after the given number of ticks have passed. 60 ticks are one second. Using a small value can slow down processing a lot while a big value keeps your application not responding to mouse clicks.

If you use this property with e.g. 6 as the value, you may also want to use this method in a thread so you can handle mouse events or let REALbasic redraw a progressbar.

(Read and Write property)

2.1.3 Constants

Dither90Halftone6x6Matrix = 5

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherCluster3Matrix = 8

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherCluster4Matrix = 9

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherCluster8Matrix = & h0000000A

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherLines4x4Matrix = 4

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherMagic2x2Matrix = 1

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherMagic4x4Matrix = 2

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherOrdered4x4Matrix = 3

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherOrdered6x6Matrix = 6

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

DitherOrdered8x8Matrix = 7

Plugin Version: 9.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the dither modes for the DitherFilter method.

ImageFormat1of3 = & h0000000F

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Notes:

This is the imageformat to use if you target only a gray channel in a RGB picture in memory. Targets the first byte with pixelsize=3.

ImageFormat1of4 = & h00000012

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Notes:

This is the imageformat to use if you target only a gray channel in a RGB picture in memory. Targets the first byte with pixelsize=4.

ImageFormat2of3 = & h00000010

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Notes:

This is the imageformat to use if you target only a gray channel in a RGB picture in memory. Targets the second byte with pixelsize=3.

ImageFormat2of4 = & h00000013

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Notes:

This is the imageformat to use if you target only a gray channel in a RGB picture in memory. Targets the second byte with pixelsize=4.

ImageFormat3of3 = & h00000011

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Notes:

This is the imageformat to use if you target only a gray channel in a RGB picture in memory. Targets the third byte with pixelsize=3.

ImageFormat3of4 = & h00000014

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Notes:

This is the imageformat to use if you target only a gray channel in a RGB picture in memory. Targets the third byte with pixelsize=4.

ImageFormat4of4 = & h00000015

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Notes:

This is the imageformat to use if you target only a gray channel in a RGB picture in memory. Targets the forth byte with pixelsize=4.

ImageFormatABGR = 9

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

ImageFormatACMYK = & h00000019

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatACMYK)
```

ImageFormatAG = & h0000000D

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatAG)
```

ImageFormatAKYMC = & h0000001E

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatAKYMC)
```

ImageFormatAofABGR = & h00000012

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatAofABGR)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatAofARGB = & h00000012

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatAofARGB)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatAofBGRA = & h00000015

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatAofBGRA)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatAofRGBA = & h00000015

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatAofRGBA)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatARGB = 4

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatARGB)
```

ImageFormatBGR = 6

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBGR)
```

ImageFormatBGRA = 7

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBGRA)
```

ImageFormatBGRX = 8

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBGRX)
```

ImageFormatBofABGR = & h00000013

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBofABGR)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatBofARGB = & h00000015

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBofARGB)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatBofBGR = & h0000000F

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBofBGR)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatBofBGRA = & h00000012

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBofBGRA)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatBofRGB = & h00000011

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBofRGB)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatBofRGBA = & h00000014

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatBofRGBA)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatBuffer = & h00000016

Plugin Version: 10.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The image format for picture objects which are used for data storage.

Notes: This format is for PixelSize = 1 and no channels.

ImageFormatCMYK = & h00000017

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatCMYK)
```

ImageFormatCMYKA = & h00000018

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatCMYKA)
```

ImageFormatCMYKX = & h0000001A

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatCMYKX)
```

ImageFormatG = & h0000000B

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatG)
```

ImageFormatGA = & h0000000C

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGA)
```

ImageFormatGofABGR = & h00000014

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGofABGR)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatGofARGB = & h00000014

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
// create a grayscale picture with 4 bytes per pixel
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGofARGB)

// fill top left pixels white
p.FillRect(0,0,10,10,255)

Backdrop=p.CopyPicture
Title=str(p.PixelSize)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatGofBGR = & h00000010

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGofBGR)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatGofBGRA = & h00000013

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGofBGRA)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatGofRGB = & h00000010

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGofRGB)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatGofRGBA = & h00000013

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatGofRGBA)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatKYMC = & h0000001C

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatKYMC)
```

ImageFormatKYMCA = & h0000001D

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatKYMCA)
```

ImageFormatKYMCA = & h0000001D

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatKYMCA)
```

ImageFormatRGB = 1

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGB)
```

ImageFormatRGBA = 2

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```

dim fSource as FolderItem = SpecialFolder.Desktop.Child("test.png") // some png with alpha
dim oPNGInput as new PNGReaderMBS

If oPNGInput.OpenFile(fSource) Then
If oPNGInput.ApplyOptions(0) Then

dim imgSource as New PictureMBS(oPNGInput.Width, oPNGInput.Height, PictureMBS.ImageFormatRGBA)

' Read row by row the file and puts it in a PictureMBS instance

dim nMax as integer = oPNGInput.Height - 1
For nInd as integer = 0 To nMax
imgSource.RowInFormat(nInd, PictureMBS.ImageFormatRGBA, true) = oPNGInput.ReadRow()
Next

' show only alpha/mask channel
Backdrop=imgSource.AlphaChannel.CopyPicture

' show Picture without mask
Backdrop=imgSource.CopyPicture

' show picture with mask
Backdrop=imgSource.CopyPictureWithMask

End If
End If

```

ImageFormatRGBX = 3

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRGBX)
```

ImageFormatRofABGR = & h00000015

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRofABGR)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatRofARGB = & h00000013

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRofARGB)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatRofBGR = & h00000011

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRofBGR)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatRofBGRA = & h00000014

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRofBGRA)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatRofRGB = & h0000000F

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRofRGB)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatRofRGBA = & h00000012

Plugin Version: 9.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatRofRGBA)
```

Notes: This is the imageformat to use if you target only a gray channel in a RGB picture in memory.

ImageFormatScaling1 = & h00000021

Plugin Version: 11.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scaling image formats.

Notes:

Used for the temporary picture while scaling.

One Byte per Pixel.

ImageFormatScaling2 = & h00000022

Plugin Version: 11.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scaling image formats.

Notes:

Used for the temporary picture while scaling.
2 Bytes per Pixel.

ImageFormatScaling3 = & h00000023

Plugin Version: 11.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scaling image formats.

Notes:

Used for the temporary picture while scaling.
3 Bytes per Pixel.

ImageFormatScaling4 = & h00000024

Plugin Version: 11.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scaling image formats.

Notes:

Used for the temporary picture while scaling.
4 Bytes per Pixel.

ImageFormatScaling5 = & h00000025

Plugin Version: 11.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scaling image formats.

Notes:

Used for the temporary picture while scaling.
5 Bytes per Pixel.

ImageFormatScaling6 = & h00000026

Plugin Version: 11.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scaling image formats.

Notes:

Used for the temporary picture while scaling.
6 Bytes per Pixel.

ImageFormatScaling7 = & h00000027

Plugin Version: 11.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scaling image formats.

Notes:

Used for the temporary picture while scaling.
7 Bytes per Pixel.

ImageFormatScaling8 = & h00000028

Plugin Version: 11.3 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scaling image formats.

Notes:

Used for the temporary picture while scaling.
8 Bytes per Pixel.

ImageFormatUnknown = 0

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

ImageFormatXBGR = & h0000000A

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatXBGR)
```

ImageFormatXCMYK = & h0000001B

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatXCMYK)
```

ImageFormatXKYMC = & h00000020

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatXKYMC)
```

ImageFormatXRGB = 5

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the possible image formats.

Example:

```
dim p as new PictureMBS(100,100,PictureMBS.ImageFormatXRGB)
```

ScaleBox = 2

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scale modes for the Scale function.

ScaleCubic = 7

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scale modes for the Scale function.

ScaleLanczos3 = 3

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scale modes for the Scale function.

ScaleLanczos8 = 4

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scale modes for the Scale function.

ScaleMitchell = 5

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scale modes for the Scale function.

ScalePoly3 = 6

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scale modes for the Scale function.

ScaleTriangle = 1

Plugin Version: 8.7 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the scale modes for the Scale function.

2.2 class PictureFactoryMBS

class PictureFactoryMBS

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The factory class for pictures.

Notes:

This class gives you a global event where you can provide your own pictures.

Whenever the plugin needs a new PictureMBS object for the result of a function or for temporary storage, you can provide one.

This is mainly for the case where you use virtual memory or you want to reuse pictures.

2.2.1 Methods

SetFactory(factory as PictureFactoryMBS)

Plugin Version: 9.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Sets the global factory object.

Notes: You can set to nil to delete the existing factory.

2.2.2 Events

NewPictureMBS(Width as integer, Height as integer, ImageFormat as integer) as PictureMBS

Plugin Version: 9.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The factory event.

Example:

```
function NewPictureMBS(Width as integer, Height as integer, ImageFormat as integer) as PictureMBS
return new PictureMBS(width, height, ImageFormat)
end function
```

Notes:

This event is called whenever a picture is requested.

Return an picture you created.

The plugin will check the Valid property for this picture and use it only if Valid is true.

Chapter 3

List of all classes

- PictureFactoryMBS 133
- PictureMBS 11