

# w3 Image 2.0

## Reference Manual



# Contents

CONTENTS .....	2
COPYRIGHT .....	4
INTRODUCTION .....	5
<i>Image::SampleFunction</i> .....	5
REFERENCE .....	6
METHODS .....	6
<i>Image::CreateEmptySurface</i> .....	6
<i>Image::SetPixel</i> .....	6
<i>Image::GetPixel</i> .....	7
<i>Image::DrawLine</i> .....	7
<i>Image::DrawEllipse</i> .....	7
<i>Image::DrawRect</i> .....	8
<i>Image::DrawText</i> .....	9
<i>Image::CreatePen</i> .....	9
<i>Image::CreateSolidBrush</i> .....	10
<i>Image::CreateFont</i> .....	10
<i>Image::SetPen</i> .....	11
<i>Image::SetBrush</i> .....	11
<i>Image::SetFont</i> .....	12
<i>Image::LoadImage</i> .....	12
<i>Image::LoadImageFromStream 2.0</i> .....	13
<i>Image::LoadImageFromUrl 2.0</i> .....	14
<i>Image::SaveImage</i> .....	14
<i>Image::StreamImage</i> .....	15
<i>Image::Clone</i> .....	17
<i>Image::FloodFill</i> .....	18
<i>Image::StretchBlt</i> .....	18
<i>Image::StretchBltExt</i> .....	19
<i>Image::Crop</i> .....	21
<i>Image::Scale / Image::ScaleImage(Visual Basic)</i> .....	21
<i>Image::Stretch</i> .....	22
<i>Image::Rotate</i> .....	22
<i>Image::Flip</i> .....	23
<i>Image::GetTextHeight</i> .....	23
<i>Image::GetTextWidth</i> .....	23
<i>Image::GetFontFamiliesName</i> .....	24
<i>Image::CreateColor/CreateColour</i> .....	24
<i>Image::CreateColorRGB/CreateColourRGB</i> .....	25
<i>Image::GrayScale 2.0</i> .....	25
<i>Image::Threshold 2.0</i> .....	25
<i>Image::Negative 2.0</i> .....	26
<i>Image::RotateCenter 2.0</i> .....	26
<i>Image::Brightness 2.0</i> .....	27
<i>Image::Contrast 2.0</i> .....	27
<i>Image::ShiftRGB 2.0</i> .....	27
<i>Image::LockMemory (IDataLock interface)</i> .....	28
<i>Image::UnlockMemory (IDataLock interface)</i> .....	28
<i>Font::GetKernAmount</i> .....	29

## w3 Image 2.0 Reference

<i>Font::GetFirstKernChar</i> .....	29
<i>Font::GetSecondKernChar</i> .....	29
PROPERTIES .....	30
<i>Image::version 2.0</i> .....	30
<i>Image::getFontFamiliesCount</i> .....	30
<i>Image::width</i> .....	30
<i>Image::height</i> .....	30
<i>Image::bkColor/bkColour</i> .....	31
<i>Image::bkMode</i> .....	31
<i>Pen::width</i> .....	31
<i>Pen::color/colour</i> .....	32
<i>Pen::style</i> .....	32
<i>Brush::color/colour</i> .....	32
<i>Font::name</i> .....	32
<i>Font::height</i> .....	33
<i>Font::width</i> .....	33
<i>Font::weight</i> .....	33
<i>Font::orientation</i> .....	33
<i>Font::color/colour</i> .....	33
<i>Font::italic</i> .....	34
<i>Font::underlined</i> .....	34
<i>Font::antialiasing</i> .....	34
<i>Font::kerningpairscount</i> .....	34
<i>Color::value (default in the interface)</i> .....	35
<i>Color::red</i> .....	35
<i>Color::green</i> .....	35
<i>Color::blue</i> .....	35
<i>Color::webcolorname/webcolourname</i> .....	36
<i>Color::webcolor/webcolour</i> .....	36
<b>APPENDIX B – IMAGE FORMATS</b> .....	<b>37</b>

# Copyright

We have done our best to ensure that the information in this manual is both accurate and useful; however, please be aware that errors may exist, and that Dimac Development / Duplo AB (hereafter referred to as Duplo) does not give any guarantees concerning the accuracy of the information here or in the use to which it may be put.

Duplo may have patents and/or pending patent applications covering subject matter in this document.

Copyright © 2004 Duplo AB. All rights reserved.

This document may only be reproduced (in whole or part), copied, photocopied, translated, or converted to any electronic or machine readable form with the prior permission of Duplo.

Dimac Development  
Duplo AB  
Prästgatan 12  
252 24 Helsingborg  
Sweden

Phone: +46 42 35 94 00  
Fax: +46 42 15 80 50  
Homepage: <http://www.dimac.net>  
E-mail: [info@dimac.net](mailto:info@dimac.net)  
Support: [support@dimac.net](mailto:support@dimac.net)

If any problems occur using this application, please visit our web site  
<http://www.dimac.net>.

# Introduction

This is the Technical Reference Manual for w3 Image 2.0. This is how you make magic happen with w3 Image. For explanations of what you can do and how the component works, please consult the Usage Manual and the samples bundled with the installation.

Using w3 Image requires some programming skill, and an understanding of the w3 Image object model.

How to read this document? Take a look at the sample below.

## Image::SampleFunction

<i>Name</i>	SampleFunction (property or method name)
<i>Description</i>	What does this property or method do?
<i>Type</i>	Which type of thing am I. Method or Property.
<i>Object</i>	Which object type do I belong to.
<i>Interface</i>	<pre>VARIANT_BOOL SampleFunction(     long aValue,     bstr aNotherValue);  // describes how the Function is implemented, which parameters it expects and which datatypes.</pre>

### Parameters

<code>aValue</code>	What does this parameter do.
<code>aNotherValue</code>	What does this parameter do.

*Return value* What the function returns, if anything.

### Example

#### JScript

```
var test=imageobj.SampleFunction (100,"a text");
```

#### VBScript

```
test=imageobj.SampleFunction (200,"another text")
```

# Reference

## Methods

### Image::CreateEmptySurface

*Name* CreateEmptySurface  
*Description* Creates a surface that is used when drawing.  
*Type* Method  
*Object* Image  
*Interface* VARIANT\_BOOL CreateEmptySurface(  
    long iWidth,  
    long iHeight);

#### Parameters

*iWidth* The width of the image to be created.  
*iHeight* The height of the image to be created.

*Return value* Returns true or false depending if the operation was successful.

#### Example

##### JScript

```
var imageobj=Server.CreateObject("w3Image.Image");  
var test=imageobj.CreateEmptySurface(100,100);
```

##### VBScript

```
Set imageobj=Server.CreateObject("w3Image.Image")  
test=imageobj.CreateEmptySurface(100,100)
```

### Image::SetPixel

*Name* SetPixel  
*Description* Sets a pixel value in the image.  
*Object* Image  
*Type M* Method  
*Interface* SetPixel(  
    long iX,  
    long iY,  
    long iColor);

#### Parameters

*iX* Specifies the x-coordinate to be set.  
*iY* Specifies the y-coordinate to be set.  
*iColor* Specifies the color used to paint the point.

*Return value* <none>

#### Example

##### JScript

```
imageobj.SetPixel(10,10,0xFF0000);
```

*VBScript*

```
imageobj.SetPixel 10,10,&H00FF0000&
```

## Image::GetPixel

*Name* GetPixel

*Description* Gets a pixel value in the image.

*Object* Image

*Type* Method

*Interface*

```
long GetPixel(  
    long iX,  
    long iY);
```

*Parameters*

*iX* Specifies the x-coordinate to be examined.

*iY* Specifies the y-coordinate to be examined.

*Return value* Retrieves the RGB value of the pixel at the point specified by x and y.

*Example*

*JScript*

```
var color = imageobj.GetPixel(10,10);
```

*VBScript*

```
color = imageobj.GetPixel(10,10)
```

## Image::DrawLine

*Name* DrawLine

*Description* Draws a line between two points in the image.

Uses the selected pen object to perform the operation.

*Object* Image

*Type* Method

*Interface*

```
DrawLine(  
    long iX1,  
    long iY1,  
    long iX2,  
    long iY2);
```

*Parameters*

*iX1* Specifies the x-coordinate of the start point.

*iY1* Specifies the y-coordinate of the start point.

*iX2* Specifies x-coordinate of the end point.

*iY2* Specifies y-coordinate of the end point.

*Return value* <none>

*Example*

*JScript*

```
imageobj.DrawLine(10,10,100,100);
```

*VBScript*

```
imageobj.DrawLine 10,10,100,100
```

## Image::DrawEllipse

## w3 Image 2.0 Reference

<i>Name</i>	DrawEllipse
<i>Description</i>	Draws an ellipse from the coordinates of the upper-left corner to the lower-right corner of the ellipse's bounding rectangle. Uses the selected pen and brush object to perform the operation.
<i>Object</i>	Image
<i>Type</i>	Method
<i>Interface</i>	<pre>DrawEllipse(     long iLeft,     long iTop,     long iRight,     long iBottom);</pre>
<i>Parameters</i>	
<i>iLeft</i>	Specifies the left edge coordinate of the ellipse.
<i>iTop</i>	Specifies the top edge coordinate of the ellipse.
<i>iRight</i>	Specifies the right edge coordinate of the ellipse.
<i>iBottom</i>	Specifies the bottom edge coordinate of the ellipse.
<i>Return value</i>	<none>
<i>Example</i>	
<i>JScript</i>	<pre>imageobj.DrawEllipse(10,10,100,100);</pre>
<i>VBScript</i>	<pre>imageobj.DrawEllipse 10,10,100,100</pre>

## Image::DrawRect

<i>Name</i>	DrawRect
<i>Description</i>	Draws a rectangle from the coordinates of the upper-left corner to the lower-right corner of the rectangle. Uses the selected pen and brush object to perform the operation.
<i>Object</i>	Image
<i>Type</i>	Method
<i>Interface</i>	<pre>DrawRect(     long iLeft,     long iTop,     long iRight,     long iBottom);</pre>
<i>Parameters</i>	
<i>iLeft</i>	Specifies the left edge coordinate of the rectangle.
<i>iTop</i>	Specifies the top edge coordinate of the rectangle.
<i>iRight</i>	Specifies the right edge coordinate of the rectangle.
<i>iBottom</i>	Specifies the bottom edge coordinate of the rectangle.
<i>Return value</i>	<none>
<i>Example</i>	
<i>JScript</i>	<pre>imageobj.DrawRect(10,10,100,100);</pre>

VBScript

```
imageobj.DrawRect 10,10,100,100
```

## Image::DrawText

*Name* DrawText

*Description* Draws a text with help of the selected font and the coordinates.

*Object* Image

*Type* Method

*Interface* DrawText (  
    BSTR bstrText,  
    long iLeft,  
    long iTop);

*Parameters*

*bstrText* Specifies the text to draw.

*iLeft* Specifies the left edge coordinate of the text.

*iTop* Specifies the top edge coordinate of the text.

*Return value* <none>

*Example*

*JScript*

```
imageobj.DrawText ("Apa", 10, 10);
```

*VBScript*

```
imageobj.DrawText "Apa", 10, 10
```

## Image::CreatePen

*Name* CreatePen

*Description* Creates a pen object to use for drawing methods.

*Object* Image

*Type* Method

*Interface* IUnknown\* = CreatePen(  
    VARIANT vtStyle,  
    long iWidth,  
    long iColor);

*Parameters*

*vtStyle* Specifies the style to use when drawing.

**(0 or "solid")** The pen is solid.

**(1 or "dash")** The pen is dashed. This style is valid only when the pen width is one.

**(2 or "dot")** The pen is dotted. This style is valid only when the pen width is one.

**(3 or "dashdot")** The pen has alternating dashes and dots. This style is valid only when the pen width is one.

**(4 or "dashdotdot")** The pen has alternating dashes and double dots. This style is valid only when the pen width is one or less in device units.

**(5 or "invisible")** The pen is invisible.

*iWidth* Specifies the width of the pen.

*iColor* Specifies the color value of the pen.

## w3 Image 2.0 Reference

*Return value* Returns a pen object.

*Example*

*JScript*

```
var penobj = imageobj.CreatePen("solid",1, 0x0000FF);
```

*VBScript*

```
Set penobj = imageobj.CreatePen("solid",1, &H000000FF&)
```

### Image::CreateSolidBrush

*Name* CreateSolidBrush

*Description* Creates a brush object to use for drawing methods.

*Object* Image

*Type* Method

*Interface*

```
IUnknown* = CreateSolidBrush(  
    long iColor);
```

*Parameters*

*iColor* Specifies the color value of the brush.

*Return value* Returns a brush object.

*Example*

*JScript*

```
var brushobj = imageobj.CreateSolidBrush(0x00FF00);
```

*VBScript*

```
Set brushobj = imageobj.CreateSolidBrush(&H0000FF00&)
```

### Image::CreateFont

*Name* CreateFont

*Description* Creates a font object to use for text methods.

*Object* Image

*Type* Method

*Interface*

```
IUnknown* = CreateFont(  
    BSTR bstrName,  
    long iHeight,  
    long iWidth,  
    VARIANT vtWeight,  
    long iOrientation,  
    long iFontColor,  
    VARIANT_BOOL bItalic,  
    VARIANT_BOOL bUnderlined,  
    VARIANT_BOOL bAntiAliasing);
```

*Parameters*

*bstrName* Specifies the face name of the font.

*iHeight* Specifies the height of the font.

*iWidth* Specifies the width of the font. If zero is specified then the default width value is used. This value is relative to the height.

*iWeight* Specifies the font weight to be used.

(0 or "dontcare") Don't care

(1 or "thin") Thin

(2 or "extralight") Extra light

## w3 Image 2.0 Reference

(3 or "light")	Light
(4 or "normal")	Normal
(5 or "medium")	Medium
(6 or "semibold")	Semi bold
(7 or "bold")	Bold
(8 or "extrabold")	Extra bold
(9 or "heavy")	Heavy

<i>iOrientation</i>	Specifies the orientation of the font (0 - 3599).
<i>iFontColor</i>	Specifies the font color.
<i>bItalic</i>	Specifies if the font should be italicised.
<i>bUnderlined</i>	Specifies if the font should be underlined.
<i>bAntiAliasing</i>	Specifies if the font should be antialiased (not always possible).

*Return value* Returns a font object.

### *Example*

#### *JScript*

```
var fontobj = imageobj.CreateFont("Wingdings", 25,  
0,"normal", 0, 0xFFFF00, false, false, false);
```

#### *VBScript*

```
Set fontobj = imageobj.CreateFont("Wingdings", 25,  
0,"normal", 0, &H00FFFF00&, false, false, false)
```

## **Image::SetPen**

<i>Name</i>	SetPen
<i>Description</i>	Select a pen object to use for drawing methods.
<i>Object</i>	Image
<i>Type</i>	Method
<i>Interface</i>	SetPen ( IUnknown* pPen);

### *Parameters*

<i>pPen</i>	Specifies the pen object to be selected.
<i>iColor</i>	

*Return value* <none>

### *Example*

#### *JScript*

```
imageobj.SetPen(penobj);
```

#### *VBScript*

```
imageobj.SetPen penobj
```

## **Image::SetBrush**

<i>Name:</i>	SetBrush
<i>Description</i>	Selects a brush object to use for drawing methods.
<i>Object</i>	Image
<i>Type</i>	Method

## w3 Image 2.0 Reference

*Interface*      `SetBrush(  
                  IUnknown* pBrush);`

*Parameters*

`pBrush`        Specifies the brush object to be selected.

*Return value*   <none>

*Example*

*JScript*

```
imageobj.SetBrush(brushobj);
```

*VBScript*

```
imageobj.SetBrush brushobj
```

### **Image::SetFont**

*Name*            `SetFont`

*Description*    Selects a font object to use for text methods.

*Object*          `Image`

*Type*            `Method`

*Interface*      `SetFont(  
                  IUnknown* pFont);`

*Parameters*

`pFont`          Specifies the font object to be selected.

*Return value*   <none>

*Example*

*JScript*

```
Imageobj.SetFont(fontobj);
```

*VBScript*

```
Imageobj.SetFont fontobj
```

### **Image::LoadImage**

*Name*            `LoadImage`

*Description*    Loads an image from file.

*Object*          `Image`

*Type*            `Method`

*Interface*      `VARIANT_BOOL LoadImage(  
                  BSRT bstrFile);`

*Parameters*

`bstrFile`       Specifies the path and the filename to the file.

*Return value*   Returns true or false depending if the operation was successfully.

*Example*

*JScript*

```
var test = imageobj.LoadImage("c:\\temp\\w3image.jpg");
```

*VBScript*

```
test = imageobj.LoadImage("c:\temp\w3image.jpg")
```

#### *Note:*

In version 2.0, the imageobject will inherit properties from the loaded image, for example transparency and background-color. This might lead to some issues when loading non-transparent images since all images per

default were transparent in earlier versions. This can be resolved easily by setting `image.bkMode = 1` after loading non-transparent images.

## Image::LoadImageFromStream 2.0

*Name* LoadImageFromStream  
*Description* Loads an image from stream.  
*Type* Method  
*Object* Image  
*Interface* VARIANT\_BOOL LoadImageFromStream(  
IUnknown \*pUnkStream);

### Parameters

`pUnkStream` Specifies the input stream.

*Return value* Returns true or false depending if the operation was successful.

**Warning:** Loading to or saving to an `adodb.stream` requires MDAC 2.6 or greater. It will not work with 2.5 and lower!

### Note:

In version 2.0, the `imageobject` will inherit properties from the loaded image, for example transparency and background-color. This might lead to some issues when loading non-transparent images since all images per default were transparent in earlier versions. This can be resolved easily by setting `image.bkMode = 1` after loading non-transparent images.

### Example

#### JScript

```
// Create Stream Object
var FS = Server.CreateObject("ADODB.Stream");

// Create Image Object
var ImageObj = Server.CreateObject("w3Image.Image");

FS.Type = 1;
FS.Open();

FS.LoadFromFile("D:\\www\\test.jpg");
ImageObj.LoadImageFromStream(FS);

FS.Close();

ImageObj.StreamImage(Response, "JPG", 24);
```

#### VBScript

```
Dim FS
Dim ImageObj

' Create Stream Object
Set Fs = Server.CreateObject("ADODB.Stream")
```

## w3 Image 2.0 Reference

```
` Create Image Object
Set ImageObj = Server.CreateObject("w3Image.Image")

FS.Type = 1
FS.Open

FS.LoadFromFile "D:\www\test.jpg"
ImageObj.LoadImageFromStream FS

FS.Close

ImageObj.StreamImage Response, "JPG", 24
```

### Image::LoadImageFromUrl 2.0

<i>Name</i>	LoadImageFromUrl
<i>Description</i>	Loads an image from an url.
<i>Type</i>	Method
<i>Object</i>	Image
<i>Interface</i>	VARIANT_BOOL LoadImageFromUrl( bstr bstrUrl);

#### Parameters

<i>bstrUrl</i>	Specifies the internet location of the image file to be downloaded.
<i>Return value</i>	Returns true or false depending if the operation was successful.

#### Note:

In version 2.0, the imageobject will inherit properties from the loaded image, for example transparency and background-color. This might lead to some issues when loading non-transparent images since all images per default were transparent in earlier versions. This can be resolved easily by setting `image.bkMode = 1` after loading non-transparent images.

#### Example

##### JScript

```
test=imageobj.LoadImageFromUrl(  
"http://www.test.com/test.jpg");
```

##### VBScript

```
test=imageobj.LoadImageFromUrl(  
"http://www.test.com/test.jpg")
```

### Image::SaveImage

<i>Name</i>	SaveImage
<i>Description</i>	Saves an image to a file.
<i>Object</i>	Image
<i>Type</i>	Method
<i>Interface</i>	VARIANT_BOOL SaveImage( 

## w3 Image 2.0 Reference

```
BSTR bstrFile,  
VARIANT vtType,  
long iColorDepth,  
VARIANT vtParam1, // Optional  
VARIANT vtParam2); // Optional
```

### Parameters

**bstrFile** Specifies the path and the filename to the file.

**vtType** Specifies the output format:

(0 or "**BMP**") - BMP

(1 or "**JPG**") - JPG

(2 or "**PNG**") - PNG

(3 or "**GIF**") - GIF

(4 or "**TIF**") - TIF

(5 or "**ICO**") - ICO

(6 or "**TGA**") - TGA

(7 or "**WBMP**") - WBMP

**iColorDepth** Specifies the colordepth of the output file.

**vtParam1**

In case image is JPEG, use this param for jpeg-quality (default is 90 (90%)), and if image is 1, 4 or 8 bit use this param to specify which palette you want to use.

(0 or "**firstfound**") - First Found

(1 or "**grayscale**") - Grayscale

(2 or "**rainbow**") - Rainbow

(3 or "**nearest**") - Nearest

More information about these palette values can be found in the Usage Manual

Valid values: **BSTR** or **long**.

**vtParam2** Specifies an extra optional parameter.

**Return value** Returns true or false depending if the operation was successful.

### Example

#### JScript

```
var test = imageobj.SaveImage("c:\\temp\\w3image.jpg",  
"BMP", 24);
```

#### VBScript

```
test = imageobj.SaveImage("c:\\temp\\w3image.jpg", "BMP", 24)  
test = imageobj.SaveImage("c:\\temp\\w3image.jpg", "GIF", 8,  
"grayscale")
```

## Image::StreamImage

**Name** StreamImage

**Description** Streams image data directly to a specified stream object that supports the IStream or IResponse interface. See Usage Manual for more information.

**Object** Image

**Type** Method

## w3 Image 2.0 Reference

*Interface*      `VARIANT_BOOL StreamImage(  
                          IUnknown* pUnkResponse,  
                          VARIANT vtType,  
                          long iColorDepth  
                          VARIANT vtParam1, // Optional  
                          VARIANT vtParam2); // Optional`

### *Parameters*

`pUnkResponse` Specifies the stream object which should be used to stream the data.

`vtType` Specifies the output format:

(0 or "BMP") - BMP

(1 or "JPG") - JPG

(2 or "PNG") - PNG

(3 or "GIF") - GIF

(4 or "TIF") - TIF

(5 or "ICO") - ICO

(6 or "TGA") - TGA

(7 or "WBMP") - WBMP\*

\* Requires that the client can handle wbmp images. Most web browsers can't.

`IColorDepth` Specifies the colordepth of the output stream.

`vtParam1`

In case image is JPEG, use this param for jpeg-quality (default is 90 (90%)), and if image is 1, 4 or 8 bit use this param to specify which palette you want to use.

(0 or "firstfound") - First Found (default)

(1 or "grayscale") - Grayscale

(2 or "rainbow") - Rainbow

(3 or "nearest") - Nearest

More information about these palette values can be found in the Usage Manual

Valid values: `BSTR` or `long`.

`vtParam2` Specifies an extra optional parameter.

*Return value* Returns true or false depending if the operation was successful.

*Warning:*      **Loading to or saving to an adodb.stream requires MDAC 2.6 or greater.  
It will not work with 2.5 and lower!**

### *Example*

#### *JScript*

```
var test = imageobj.StreamImage(Response, "PNG", 8,  
"rainbow");
```

#### *VBScript*

```
test = imageobj.StreamImage(Response, "PNG", 8)
```

## Image::Clone

<i>Name</i>	Clone
<i>Description</i>	Clones an image object.
<i>Object</i>	Image
<i>Type</i>	Method
<i>Interface</i>	<code>IUnknown** Clone( );</code>
<i>Parameters</i>	<none>
<i>Return value</i>	Returns the cloned image object.
<i>Example</i>	
<i>JScript</i>	<code>var imageobj2 = imageobj.Clone();</code>
<i>VBScript</i>	<code>imageobj2 = imageobj.Clone</code>

## Image::FloodFill

*Name* FloodFill  
*Description* Fills an area in the image with the current selected brush.  
*Object* Image  
*Type* Method  
*Interface* FloodFill(  
    long iX,  
    long iY,  
    long iColor);

### Parameters

*iX* Specifies the x-coordinate of the point where the filling is to start.  
*iY* Specifies the y-coordinate of the point where the filling is to start.  
*iColor* Specifies the color of the boundary or the area to be filled.

*Return value* <none>

### Example

#### JScript

```
imageobj.FloodFill(10,10,0xFF0088);
```

#### VBScript

```
imageobj.FloodFill 10,10,&HFF0088
```

## Image::StretchBlt

*Name* StretchBlt  
*Description* Blits data from one part of the image to another.  
*Object* Image  
*Type* Method  
*Interface* StretchBlt(  
    long iXDest,  
    long iYDest,  
    long iDestWidth,  
    long iDestHeight,  
    long iXSrc,  
    long iYSrc,  
    long iSrcWidth,  
    long iSrcHeight,  
    VARIANT vtRasterOperation);

### Parameters

*iXDest* Specifies the x-coordinate of the upper-left corner of the destination rectangle.  
*iYDest* Specifies the y-coordinate of the upper-left corner of the destination rectangle.  
*iDestWidth* Specifies the width of the destination rectangle.  
*iDestHeight* Specifies the height of the destination rectangle.  
*iXSrc* Specifies the x-coordinate of the upper-left corner of the source rectangle.  
*iYSrc* Specifies the y-coordinate of the upper-left corner of the source rectangle.  
*iSrcWidth* Specifies the width of the source rectangle.  
*iSrcHeight* Specifies the height of the source rectangle.

vtRaster-  
Operation

Specifies the raster operation:

(**1** or "**dstinvert**") - Inverts the destination rectangle.

(**2** or "**notsrccopy**") - Copies the inverted source rectangle to the destination.

(**3** or "**mergepaint**") - Merges the colors of the inverted source rectangle with the colors of the destination rectangle by using the Boolean OR operator.

(**5** or "**notsrcerase**") - Combines the colors of the source and destination rectangles by using the Boolean OR operator and then inverts the resultant color.

(**9** or "**srcand**") - Combines the colors of the source and destination rectangles by using the Boolean AND operator.

(**10** or "**srccopy**") - Copies the source rectangle directly to the destination rectangle.

(**11** or "**srcerase**") - Combines the inverted colors of the destination rectangle with the colors of the source rectangle by using the Boolean AND operator.

(**12** or "**srcinvert**") - Combines the colors of the source and destination rectangles by using the Boolean XOR operator.

(**13** or "**srcpaint**") - Combines the colors of the source and destination rectangles by using the Boolean OR operator.

*Return value* <none>

*Example*

*JScript*

```
Imageobj.StretchBlt (10,10,20,20,40,40,20,20,"srccopy");
```

*VBScript*

```
Imageobj.StretchBlt 10,10,20,20,40,40,20,20,"srccopy"
```

## Image::StretchBltExt

*Name* StretchBltExt

*Description* Blits data from one object into another.

*Object* Image

*Type* Method

*Interface*

```
StretchBltExt(  
    IUnknown* pUnkDest,  
    long iXDest,  
    long iYDest,  
    long iDestWidth,  
    long iDestHeight,  
    long iXSrc,  
    long iYSrc,  
    long iSrcWidth,  
    long iSrcHeight,  
    VARIANT vtRasterOperation // Optional  
    VARIANT vtAlpha); // Optional
```

*Parameters*

**pUnkDest** Specifies the destination object which the data should be blitted.

## w3 Image 2.0 Reference

<code>iXDest</code>	Specifies the x-coordinate of the upper-left corner of the destination rectangle.
<code>iYDest</code>	Specifies the y-coordinate of the upper-left corner of the destination rectangle.
<code>iDestWidth</code>	Specifies the width of the destination rectangle.
<code>iDestHeight</code>	Specifies the height of the destination rectangle.
<code>iXSrc</code>	Specifies the x-coordinate of the upper-left corner of the source rectangle.
<code>iYSrc</code>	Specifies the y-coordinate of the upper-left corner of the source rectangle.
<code>iSrcWidth</code>	Specifies the width of the source rectangle.
<code>iSrcHeight</code>	Specifies the height of the source rectangle.
<code>vtRasterOperation</code>	Specifies the raster operation: ( <b>1</b> or " <b>dstinvert</b> ") - Inverts the destination rectangle. ( <b>2</b> or " <b>notsrccopy</b> ") - Copies the inverted source rectangle to the destination. ( <b>3</b> or " <b>mergepaint</b> ") - Merges the colors of the inverted source rectangle with the colors of the destination rectangle by using the Boolean OR operator. ( <b>5</b> or " <b>notsrcerase</b> ") - Combines the colors of the source and destination rectangles by using the Boolean OR operator and then inverts the resultant color. ( <b>9</b> or " <b>srcand</b> ") - Combines the colors of the source and destination rectangles by using the Boolean AND operator. ( <b>10</b> or " <b>srccopy</b> ") - Copies the source rectangle directly to the destination rectangle. ( <b>11</b> or " <b>srcerase</b> ") - Combines the inverted colors of the destinationrectangle with the colors of the source rectangle by using the Boolean AND operator. ( <b>12</b> or " <b>srcinvert</b> ") - Combines the colors of the source and destination rectangles by using the Boolean XOR operator. ( <b>13</b> or " <b>srcpaint</b> ") - Combines the colors of the source and destination rectangles by using the Boolean OR operator. ( <b>15</b> or " <b>transparent</b> ") - Merges the source and the destination rectangle. This by using the background color as the transparent color in the source image (bkColor). The background mode is here ignored (bkMode). ( <b>16</b> or " <b>alpha</b> ") - Merges the source and destinationen rectangle using the alpha value (0 - 255). ( <b>17</b> or " <b>alphatransparent</b> ") - Combination of the transparent and alpha blit operation. Useful e.g. when merging a logotype into an image. <b>1.3</b>
<code>vtAlpha</code>	Specifies the alpha value (0-255) used when performing an alpha blending operation

## w3 Image 2.0 Reference

*Return value* <none>

*Example*

*JScript*

```
imageobj.StretchBltExt (destobj, 10, 10, 20, 20, 10, 10, 30, 30, 9);
```

*VBScript*

```
imageobj.StretchBltExt destobj, 10, 10, 20, 20, 10, 10, 30, 30, 9
```

### **Image::Crop**

*Name* Crop

*Description* Crops the image to the rectangle specified.

*Object* Image

*Type* Method

*Interface*

```
Crop(  
    long iLeft,  
    long iTop,  
    long iWidth,  
    long iHeight);
```

*Parameters*

*ILeft* Specifies the upper-left corner of the rectangle.

*ITop* Specifies the top of the rectangle.

*iWidth* Specifies the width of the rectangle.

*iHeight* Specifies the height of the rectangle.

*Return value* <none>

*Example*

*JScript* `imageobj.Crop(10, 10, 100, 100);`

*VBScript* `imageobj.Crop 10, 10, 100, 100`

### **Image::Scale / Image::ScaleImage(Visual Basic)**

*Name* Scale / ScaleImage

*Description* Scales the image by a percent value.

*Object* Image

*Type* Method

*Interface*

```
Scale(  
    VARIANT vtVal  
    VARIANT vtInterPolate); // Optional
```

*Parameters*

*vtVal* Specifies the scale factor in percent.

*vtInterPolate* Specifies the interpolation method, this is an optional parameter.

(**0** or "**nearest**") - Nearest Neighbour.

(**1** or "**linear**") - Linear.

(**2** or "**cubic**") - Cubic.

(**4** or "**super**") - Super Sampling. May be used when the saved image is smaller than the original image.

*Return value* <none>

*Example*

*JScript* `Imageobj.Scale(200, 0); //200 %`

*VBScript* `Imageobj.ScaleImage 200, 0`

## Image::Stretch

*Name* Stretch

*Description* Stretches the image.

*Object* Image

*Type* Method

*Interface* `Stretch(  
    long iWidth,  
    long iHeight  
    VARIANT vtInterPolate); // Optional`

### Parameters

`iWidth` Specifies the new width of the image.

`iHeight` Specifies the new height of the image.

`vtInterPolate` Specifies the interpolation method, this is an optional parameter:

(0 or "**nearest**") - Nearest Neighbour.

(1 or "**linear**") - Linear.

(2 or "**cubic**") - Cubic.

(4 or "**super**") - Super Sampling. May be used when the saved image is smaller than the original image.

*Return value* <none>

### Example

*JScript* `Imageobj.Stretch(200, 300,"linear") ;`

*VBScript* `Imageobj.Stretch 200, 300, "linear"`

## Image::Rotate

*Name* Rotate

*Description* Rotates the image.

*Object* Image

*Type* Method

*Interface* `Rotate(  
    VARIANT vtAngle,  
    long iXCenter,  
    long iYCenter,  
    VARIANT vtInterPolate); // Optional`

### Parameters

`vtAngle` Specifies the rotation angle.

`iXCenter` Specifies the rotation point for the x-coordinate.

`iYCenter` Specifies the rotation point for the y-coordinate.

`vtInterPolate` Specifies the interpolation method, this is an optional parameter.

(0 or "**nearest**") - Nearest Neighbour.

(1 or "**linear**") - Linear.

(2 or "**cubic**") - Cubic.

(5 or "**smoothnearest**") - Smooth edge in combination with the Nearest Neighbour algorithm.

(6 or "**smoothlinear**") - Smooth edge in combination with the Linear algorithm.

(7 or "**smoothcubic**") - Smooth edge in combination with the

Cubic algorithm.

*Return value* <none>

*Example*

*JScript* `imageobj.Rotate(37.5,0,0,"smooth");`

*VBScript* `imageobj.Rotate 37.5,0,0,"smooth"`

## **Image::Flip**

*Name* Flip

*Description* Flips the image.

*Object* Image

*Type* Method

*Interface* `Flip(  
                  long vtFlag);`

*Parameters*

`vtFlag` Specifies if the image should be flipped vertical, horizontal or both:  
(0 or "nearest") - Both  
(1 or "vertical") - Vertical  
(2 or "horizontal") - Horizontal

*Return value* <none>.

*Example*

*JScript* `imageobj.Flip("vertical");`

*VBScript* `imageobj.Flip "vertical"`

## **Image::GetTextHeight**

*Name* GetTextHeight

*Description* Returns the height in pixels of a specified text.

*Object* Image

*Type* Method

*Interface* `long* GetTextHeight(  
                  BSTR bstrText);`

*Parameters*

`bstrText` Specifies the text to be calculated.

*Return value* Returns the height of the text.

*Example*

*JScript*

`var height = GetTextHeight("Lorem ipsum dolor sit amet.");`

*VBScript*

`height = GetTextHeight("Lorem ipsum dolor sit amet.")`

## **Image::GetTextWidth**

*Name* GetTextWidth

*Description* Returns the width in pixels of a specified text.

*Object* Image

*Type* Method

*Interface* `long* GetTextWidth(  
                  BSTR bstrText);`

*Parameters*

## w3 Image 2.0 Reference

**bstrText** Specifies the text to be calculated.

**Return value** Returns the width of the text.

**Example**

**JScript**

```
var width = GetTextWidth("Lorem ipsum dolor sit amet.");
```

**VBScript**

```
width = GetTextWidth("Lorem ipsum dolor sit amet.")
```

### **Image::GetFontFamiliesName** *1.3*

**Name** GetFontFamiliesName

**Description** Gets the names of the font families, specified by the index value, i.e. list all available fonts on the host machine. Use this together with getFontFamiliesCount to loop through the fonts.

**Object** Image

**Type** Method

**Interface** `BSTR* bstrName = GetFontFamiliesName(long iIndex);`

**Parameters**

**iIndex** Specifies the index value of a font family.

**Return value** Returns the font family name.

**Example**

**JScript**

```
var name = GetFontFamiliesName(10);
```

**VBScript**

```
name = GetFontFamiliesName(10)
```

### **Image::CreateColor/CreateColour**

**Name** CreateColor/CreateColour

**Description** Creates a color object.

**Object** Image

**Type** Method

**Interface** `IUnknown** = CreateColor(  
VARIANT vtParam);`

**Parameters**

**vtParam1** Specifies the color description value.  
Valid values: `BSTR` or `long`.

**Return value** The color object.

**Example**

**JScript**

**Example1(webcolorname):**

```
colorobj = imageobj.CreateColor("black");
```

**Example2 (webcolor):**

```
colorobj = imageobj.CreateColor("#A4F012");
```

**VBScript**

**Example1(webcolorname):**

```
Set colorobj = imageobj.CreateColor("black")
```

**Example2 (webcolor):**

```
Set colorobj = imageobj.CreateColor("#A4F012")
```

## Image::CreateColorRGB/CreateColourRGB

*Name* CreateColorRGB/CreateColourRGB

*Description* Creates a color object.

*Object* Image

*Type* Method

*Interface* IUnknown\*\* = CreateColorRGB(  
VARIANT vtParam);

*Parameters*

*iRed* Specifies the red intensity value.

*iGreen* Specifies the green intensity value.

*iBlue* Specifies the blue intensity value.

*Return value* The color object.

*Example*

*JScript* var colorobj = imageobj.CreateColorRGB(0,0,255);

*VBScript* Set colorobj = imageobj.CreateColorRGB(0,0,255)

## Image::GrayScale 2.0

*Name* GrayScale

*Description* Makes the image grayscale.

*Type* Method

*Object* Image

*Interface* VARIANT\_BOOL GrayScale();

*Parameters* <none>

*Return value* Returns true or false depending if the operation was successful.

*Example*

*JScript* var test=imageobj.GrayScale();

*VBScript* test=imageobj.GrayScale

## Image::Threshold 2.0

*Name* Threshold

*Description* Converts the image into black and white, specified by the threshold lightness value.

*Type* Method

*Object* Image

*Interface* VARIANT\_BOOL Threshold(  
long lThreshold);

*Parameters*

## w3 Image 2.0 Reference

**lThreshold** Specifies the threshold value (0-255).  
*Return value* Returns true or false depending if the operation was successful.

### Example

```
JScript      var test=imageobj.Threshold(128);  
VBScript    test=imageobj.Threshold(100)
```

## Image::Negative 2.0

*Name* Negative  
*Description* Inverts the image colors.  
*Type* Method  
*Object* Image  
*Interface* VARIANT\_BOOL Negative();  
*Parameters* <none>  
*Return value* Returns true or false depending if the operation was successful.

### Example

```
JScript      var test=imageobj.Negative();  
VBScript    test=imageobj.Negative
```

## Image::RotateCenter 2.0

*Name* RotateCenter  
*Description* Rotates clockwise the image around the center point. It rescales the image if necessary.  
*Type* Method  
*Object* Image  
*Interface* VARIANT\_BOOL RotateCenter(  
VARIANT vtAngle,  
VARIANT vtInterPolate);

### Parameters

**vtAngle** Specifies the rotation angle.  
**vtInterPolate** Specifies the interpolation method, this is an optional parameter:  
(0 or "nearest") - Nearest Neighbour.  
(1 or "linear") - Linear.  
(2 or "cubic") - Cubic.  
(5 or "smoothnearest") - Smooth edge in combination with the Nearest Neighbour algorithm.  
(6 or "smoothlinear") - Smooth edge in combination with the Linear algorithm.  
(7 or "smoothcubic") - Smooth edge in combination with the Cubic algorithm.

## w3 Image 2.0 Reference

*Return value* Returns true or false depending if the operation was successful.

### *Example*

```
JScript    var test=imageobj.RotateCenter(90,"nearest");  
VBScript  test=imageobj.RotateCenter 270 , 7
```

## **Image::Brightness 2.0**

*Name* Brightness  
*Description* Changes the brightness of the image.  
*Type* Method  
*Object* Image  
*Interface* VARIANT\_BOOL Brightness(  
long lValue);

### *Parameters*

*lValue* Specifies the brightness value (-255 to 255).  
*Return value* Returns true or false depending if the operation was successful.

### *Example*

```
JScript    var test=imageobj.Brightness(50);  
VBScript  test=imageobj.Brightness(-128)
```

## **Image::Contrast 2.0**

*Name* Contrast  
*Description* Changes the contrast of the image.  
*Type* Method  
*Object* Image  
*Interface* VARIANT\_BOOL Contrast(  
long lValue);

### *Parameters*

*lValue* Specifies the contrast value (-100 to 100).  
*Return value* Returns true or false depending if the operation was successful.

### *Example*

```
JScript    var test=imageobj.Contrast(-100);  
VBScript  test=imageobj.Contrast(100)
```

## **Image::ShiftRGB 2.0**

*Name* ShiftRGB  
*Description* Changes separately the red, green, and blue intensity values in the image.  
*Type* Method

## w3 Image 2.0 Reference

*Object* Image  
*Interface* VARIANT\_BOOL ShiftRGB(  
    long lRed,  
    long lGreen,  
    long lBlue);

*Parameters*  
*lRed* Specifies the red intensity value to be changed (-255 to 255).  
*lGreen* Specifies the green intensity value to be changed (-255 to 255).  
*lBlue* Specifies the blue intensity value to be changed (-255 to 255).

*Return value* Returns true or false depending if the operation was successful.

### *Example*

*JScript*

```
var test=imageobj.ShiftRGB(0,100,45);
```

*VBScript*

```
test=imageobj.ShiftRGB(100,56,128)
```

## **Image::LockMemory (IDataLock interface) 1.3**

*Name* LockMemory  
*Description* Locks and gives access (low level access) to the memory of the image object. The IDataLock interface is only visible in low level languages, such as C, C++. The data you can access is 24bit bitmap data. Play around with this at your own risk.

*Object* Image  
*Type* Method  
*Interface* LockMemory(DWORD\* pcbBytes, void\*\* ppBytes);

*Parameters*  
*pcbBytes* Specifies the total amount of memory bytes of the image.  
*ppBytes* Specifies the low level access memory pointer.

*Return value* <none>

*Example*  
*JScript* <not available>  
*VBScript* <not available>

## **Image::UnlockMemory (IDataLock interface) 1.3**

*Name* UnlockMemory  
*Description* Unlocks the memory of the image object.

*Object* Image  
*Type* Method  
*Interface* UnlockMemory();

*Parameters* <none>  
*Return value* <none>

*Example*  
*JScript* <not available>  
*VBScript* <not available>

## Font::GetKernAmount

*Name* GetKernAmount  
*Description* Returns the kerning amount for the specified kerningpair.  
*Object* Font  
*Type* Method  
*Interface* `long* GetKernAmount(long iIndex);`  
*Parameters*  
`iIndex` Specifies the kerning index.  
*Return value* The kerningpair amount.  
*Example*  
*JScript* `var value = fontobj.GetKernAmount(0);`  
*VBScript* `value = fontobj.GetKernAmount(0)`

## Font::GetFirstKernChar

*Name* GetFirstKernChar  
*Description* Returns the first kerningpair character.  
*Object* Font  
*Type* Method  
*Interface* `BSTR* GetFirstKernChar(long iIndex);`  
*Parameters*  
`iIndex` Specifies the kerning index.  
*Return value* The kerningpair character.  
*Example*  
*JScript* `var first = fontobj.GetFirstKernChar(0);`  
*VBScript* `first = fontobj.GetFirstKernChar(0)`

## Font::GetSecondKernChar

*Name* GetSecondKernChar  
*Description* Returns the second kerningpair character.  
*Object* Font  
*Type* Method  
*Interface* `BSTR* GeSecondKernChar(  
long iIndex);`  
*Parameters*  
`iIndex` Specifies the kerning index.  
*Return value* The kerningpair character.  
*Example*  
*JScript* `var second = fontobj.GetSecondKernChar(0);`  
*VBScript* `second = fontobj.GetSecondKernChar(0)`

## Properties

### Image::version **2.0**

*Name* version  
*Description* Gets the version number as string.  
*Object* Image  
*Interface* BSTR\* version;  
*Type* Property (Get)  
*Example*  
*JScript*  
`var version=imageobj.version;`  
*VBScript*  
`version=imageobj.version`

### Image::getFontFamiliesCount **1.3**

*Name* getFontFamiliesCount  
*Description* Counts the total amount of available fonts. See also GetFontFamiliesName.  
*Object* Image  
*Interface* long getFontFamiliesCount;  
*Type* Property (Get)  
*Example*  
*JScript* `var count = imageobj.getFontFamiliesCount;`  
*VBScript* `count = imageobj.getFontFamiliesCount`

### Image::width

*Name* width  
*Description* The width of the image.  
*Object* Image  
*Interface* long width;  
*Type* Property (Get)  
*Example*  
*JScript* `var width = imageobj.width;`  
*VBScript* `width = imageobj.width`

### Image::height

*Name* height  
*Description* The height of the image.  
*Object* Image  
*Interface* long height;  
*Type* Property (Get)  
*Example*

*JScript*        `var height = imageobj.height;`  
*VBScript*     `height = imageobj.height`

### **Image::bkColor/bkColour**

*Name*            `bkColor/bkColour`  
*Description*    The background image color. Used especially for transparent PNG or GIF images, this color will be used as the transparent color.

New in version 2.0 is that `bkColor` and `bkMode` are inherited from the loaded image. If the loaded image is transparent, `bkMode` will be set to 1 and `bkColor` will be set to the background-color of the image.

*Object*            `Image`  
*Interface*        `long bkColor;`  
*Type*              Property (Set/Get)  
*Example*  
*JScript*           `imageobj.bkColor = 0x8800FF;`  
*VBScript*         `imageobj.bkColor = &H008800FF&`

### **Image::bkMode**

*Name*            `bkMode`  
*Description*    The background mode. If this mode is set to transparent, and a GIF or PNG is generated, the color specified in `bkColor` will be used as transparent.

New in version 2.0 is that `bkColor` and `bkMode` are inherited from the loaded image. If the loaded image is transparent, `bkMode` will be set to 1 and `bkColor` will be set to the background-color of the image.

- (0) Opaque
- (1) Transparent

*Object* `Image`  
*Interface*        `long bkMode;`  
*Type*              Property (Set/Get)  
*Example*  
*JScript*           `imageobj.bkMode = 1;`  
*VBScript*         `imageobj.bkMode = 0`

### **Pen::width**

*Name*            `width`  
*Description*    The width of the pen.  
*Object*           `Pen`  
*Interface*        `long width;`  
*Type*              Property (Get)  
*Example*

## w3 Image 2.0 Reference

*JScript*        `var width = penobj.width;`  
*VBScript*     `width = penobj.width`

### **Pen::color/colour**

*Name*            `color/colour`  
*Description*    `The color value of the pen.`  
*Object*          `Pen`  
*Interface*       `long color;`  
*Type*            `Property (Get)`  
*Example*  
*JScript*        `var color = penobj.color;`  
*VBScript*      `color = penobj.color`

### **Pen::style**

*Name*            `style`  
*Description*    `The style of the pen.`  
*Object*          `Pen`  
*Interface*       `long style;`  
*Type*            `Property (Get)`  
*Example*  
*JScript*        `var style = penobj.style;`  
*VBScript*      `style = penobj.style`

### **Brush::color/colour**

*Name*            `color`  
*Description*    `The color value of the brush.`  
*Object*          `Brush`  
*Interface*       `long color;`  
*Type*            `Property (Get)`  
*Example*  
*JScript*        `var color = brushobj.color;`  
*VBScript*      `color = brushobj.color`

### **Font::name**

*Name*            `name`  
*Description*    `The face name of the font.`  
*Object*          `Font`  
*Interface*       `BSTR name;`  
*Type*            `Property (Get)`  
*Example*  
*JScript*        `var facename = fontobj.name;`  
*VBScript*      `facename = fontobj.name`

### Font::height

*Name* height  
*Description* The height in pixels of the font.  
*Object* Font  
*Interface* `long height;`  
*Type* Property (Get)  
*Example*  
*JScript* `height = fontobj.height`  
*VBScript* `var height = fontobj.height;`

### Font::width

*Name* width  
*Description* The width in pixels of the font.  
*Object* Font  
*Interface* `long width;`  
*Type* Property (Get)  
*Example*  
*JScript* `var width = fontobj.width;`  
*VBScript* `width = fontobj.width`

### Font::weight

*Name* weight  
*Description* The font weight.  
*Object* Font  
*Interface* `long weight;`  
*Type* Property (Get)  
*Example*  
*JScript* `var weight = fontobj.weight;`  
*VBScript* `weight = fontobj.weight`

### Font::orientation

*Name* orientation  
*Description* The orientation of the font.  
*Object* Font  
*Interface* `long orientation;`  
*Type* Property (Get)  
*Example*  
*JScript* `var orientation = fontobj.orientation;`  
*VBScript* `orientation = fontobj.orientation`

### Font::color/colour

*Name* color/colour  
*Description* The font color.  
*Object* Font  
*Interface* `long color;`

## w3 Image 2.0 Reference

*Type* Property (Get)  
*Example*  
*JScript* `var color = fontobj.color;`  
*VBScript* `color = fontobj.color`

### **Font::italic**

*Name* italic  
*Description* Specifies if the font is italicised.  
*Object* Font  
*Interface* `VARIANT_BOOL italic;`  
*Type* Property (Get)  
*Example*  
*JScript* `var isItalic = fontobj.italic;`  
*VBScript* `isItalic = fontobj.italic`

### **Font::underlined**

*Name* underlined  
*Description* Specifies if the font is underlined.  
*Object* Font  
*Interface* `VARIANT_BOOL underlined;`  
*Type* Property (Get)  
*Example*  
*JScript* `var isUnderlined = fontobj.underlined;`  
*VBScript* `isUnderlined = fontobj.underlined`

### **Font::antialiasing**

*Name* antialiasing  
*Description* Specifies if the font should be antialiased if possible.  
*Object* Font  
*Interface* `VARIANT_BOOL antialiasing;`  
*Type* Property (Get)  
*Example*  
*JScript* `var isAntialised = fontobj.antialiasing;`  
*VBScript* `isAntialised =fontobj.antialiasing`

### **Font::kerningpairscount**

*Name* kerningpairscount  
*Description* Gets the number of kerningpairs.  
*Object* Font  
*Interface* `long kerningpairscount;`  
*Type* Property (Get)  
*Example*  
*JScript* `var count = fontobj.kerningpairscount;`  
*VBScript* `count = fontobj.kerningpairscount`

## Color::value (default in the interface)

*Name* value  
*Description* The colorvalue.  
*Object* Color  
*Interface* long value;  
*Type* Property (Get/Set)

### Example

#### JScript

##### Example1:

```
var value = colorobj.value;
```

##### Example2:

```
var fontobj = imageobj.CreateFont("Wingdings", 25, 0, "normal",  
0, imageobj.CreateColor("snow"),false, false, false);
```

#### VBScript

##### Example 1:

```
value = colorobj.value
```

##### Example 2:

```
Set fontobj = imageobj.CreateFont("Wingdings", 25, 0, "normal",  
0, imageobj.CreateColor("snow"), false, false, false)
```

## Color::red

*Name* red  
*Description* The red intensity of the RGB color.  
*Object* Color  
*Interface* long red;  
*Type* Property (Get/Set)

### Example

#### JScript

```
var red = colorobj.red;
```

#### VBScript

```
red = colorobj.red
```

## Color::green

*Name* green  
*Description* The green intensity of the RGB color.  
*Object* Color  
*Interface* long green;  
*Type* Property (Get/Set)

### Example

#### JScript

```
var green=colorobj.green = 100;
```

#### VBScript

```
colorobj.green = 100
```

## Color::blue

*Name* blue  
*Description* The blue intensity of the RGB color.  
*Object* Color  
*Interface* long blue;  
*Type* Property (Get/Set)

### Example

*JScript*        `var blue = colorobj.blue;`  
*VBScript*      `blue = colorobj.blue`

### **Color::webcolorname/webcolourname**

*Name*            webcolorname/webcolourname  
*Description*    Specifies the webcolorname.  
*Object*          Color  
*Interface*       `BSTR webcolorname;`  
*Type*            Property (Get/Set)  
*Example*  
*JScript*        `colorobj.webcolorname = "chocolate";`  
*VBScript*      `colorobj.webcolorname = "chocolate"`

### **Color::webcolor/webcolour**

*Name*            webcolor/webcolour  
*Description*    Specifies the webcolor.  
*Object*          Color  
*Interface*       `BSTR webcolor;`  
*Type*            Property (Get/Set)  
*Example*  
*JScript*        `colorobj.webcolor = "#A312F5";`  
*VBScript*      `colorobj.webcolor = "#A312F5"`

## Appendix B – Image Formats

Format	Bitdepth	Save	Stream	Load	Transparency
<b>BMP</b>	1	X	X	X	
	4	X	X	X	
	8	X	X	X	
	24	X	X	X	

Format	Bitdepth	Save	Stream	Load	Transparency
<b>GIF</b>	1 *	X	X	X	X
	4	X	X	X	X
	8	X	X	X	X

Format	Bitdepth	Save	Stream	Load	Transparency
<b>PNG</b>	1 *	X	X	X	X
	4	X	X	X	X
	8	X	X	X	X
	24	X	X	X	
	32	X	X	X	X

Format	Bitdepth	Save	Stream	Load	Transparency
<b>JPG</b>	24	X	X	X	

Format	Bitdepth	Save	Stream	Load	Transparency
<b>ICO</b>	1 *	X	X ***	X	
	4	X	X ***	X	
	8	X	X ***	X	

Format	Bitdepth	Save	Stream	Load	Transparency
<b>TGA</b>	8	X	X ***	X	
	24	X	X ***	X	

Format	Bitdepth	Save	Stream	Load	Transparency
<b>TIFF</b>	1 *	X	X ***	X	
	4	X	X ***	X	
	8	X	X ***	X	
	24	X	X ***	X	

Format	Bitdepth	Save	Stream	Load	Transparency
<b>WBMP</b>	1 *	X	X**	X	

\* Monochrome images.

\*\* Do not stream these to normal web browsers, they do not support the WBMP image type. WAP devices do, however.

\*\*\* Streaming these images to the Response-object is not recommended, since most browsers do not support these image types. Streaming them to other objects, i.e. adodb-streams work fine, though.

**NOTE:** WBMPs larger than 127\*127 are NOT supported.

ICOs larger than 256\*256 are NOT supported.