

Image Processing (ImageMagick)

Introduction

Imagick is a native php extension to create and modify images using the ImageMagick API.

ImageMagick® is a software suite to create, edit, and compose bitmap images.. It can read, convert and write images in a variety of formats (over 100) including DPX, EXR, GIF, JPEG, JPEG-2000, PDF, PhotoCD, PNG, Postscript, SVG, and TIFF.

Copyright 1999-2007 ImageMagick Studio LLC, a non-profit organization dedicated to making software imaging solutions freely available.

Installing/Configuring

Requirements

Installation requirements on Windows

Version information does not differ from that above. There are binaries available from <http://imagemagick.org/> so that you can load this extension on *Windows* without need for a compiler.

Installation requirements on other platforms

PHP \geq 5.1.3 and ImageMagick \geq 6.2.4 is required. The amount of formats supported is by Imagick is entirely dependent upon the amount of formats supported by your ImageMagick installation. For example, Imagemagick requires ghostscript to conduct PDF operations.

Installation

This » [PECL](#) extension is not bundled with PHP.

Information for installing this PECL extension may be found in the manual chapter titled [Installation of PECL extensions](#). Additional information such as new releases, downloads, source files, maintainer information, and a CHANGELOG, can be located here: » <http://pecl.php.net/package/imagick>.

| Note |
|---|
| The official name of this extension is <i>imagick</i> . |

The DLL for this PECL extension may be downloaded from either the » [PHP Downloads](#) page or from » <http://pecl4win.php.net/>

Runtime Configuration

This extension has no configuration directives defined in *php.ini*.

Resource Types

This extension has no resource types defined.

Predefined Constants

The constants below are defined by this extension, and will only be available when the extension has either been compiled into PHP or dynamically loaded at runtime.

Colortype constants

imagick::COLOR_BLACK ([integer](#))

imagick::COLOR_BLUE ([integer](#))

imagick::COLOR_CYAN ([integer](#))
Cyan

imagick::COLOR_GREEN ([integer](#))
Green

imagick::COLOR_RED ([integer](#))
Red

imagick::COLOR_YELLOW ([integer](#))
Yellow

imagick::COLOR_MAGENTA ([integer](#))
Magenta

imagick::COLOR_OPACITY ([integer](#))
Opacity

imagick::COLOR_ALPHA ([integer](#))
Alpha

imagick::COLOR_FUZZ ([integer](#))
Fuzz

Dispose type constants

imagick::DISPOSE_UNRECOGNIZED ([integer](#))
Unrecognized

imagick::DISPOSE_UNRECOGNIZED ([integer](#))
Unrecognized

imagick::DISPOSE_UNDEFINED ([integer](#))
Undefined

imagick::DISPOSE_NONE ([integer](#))
None

imagick::DISPOSE_BACKGROUND ([integer](#))
Background

imagemagick::DISPOSE_PREVIOUS ([integer](#))

Previous

Composite Operator Constants

imagemagick::COMPOSITE_OVER ([integer](#))

Overlay one image over the next

imagemagick::COMPOSITE_IN ([integer](#))

Replaces the inside of one layer with another

imagemagick::COMPOSITE_OUT ([integer](#))

Replaces the outside of one layer with another

imagemagick::COMPOSITE_ATOP ([integer](#))

Composites the inside of one layer with the other

imagemagick::COMPOSITE_XOR ([integer](#))

The part of the source that lies outside of the destination is combined with the part of the destination that lies outside the source.

imagemagick::COMPOSITE_PLUS ([integer](#))

The source is added to the destination and replaces the destination.

imagemagick::COMPOSITE_MINUS ([integer](#))

The source is subtracted to the destination and replaces the destination.

imagemagick::COMPOSITE_ADD ([integer](#))

Deprecated

imagemagick::COMPOSITE_SUBTRACT ([integer](#))

Deprecated

imagemagick::COMPOSITE_DIFFERENCE ([integer](#))

The difference in color values. Good for comparing images.

imagemagick::COMPOSITE_BUMPMAP ([integer](#))

The same as COMPOSITE_MULTIPLY, except the source is converted to greyscale first.

imagemagick::COMPOSITE_COPY ([integer](#))

Simply place the source on top of the destination.

imagemagick::COMPOSITE_DISPLACE ([integer](#))

imagemagick::COMPOSITE_DEFAULT ([integer](#))

Montage Mode constants

imagemagick::MONTAGEMODE_FRAME ([integer](#))

imagemagick::MONTAGEMODE_UNFRAME ([integer](#))

imagick::MONTAGEMODE_CONCATENATE ([integer](#))

Style constants

imagick::STYLE_NORMAL ([integer](#))

imagick::STYLE_ITALIC ([integer](#))

imagick::STYLE_OBLIQUE ([integer](#))

imagick::STYLE_ANY ([integer](#))

Filter constants

imagick::FILTER_UNDEFINED ([integer](#))

imagick::FILTER_POINT ([integer](#))

imagick::FILTER_BOX ([integer](#))

imagick::FILTER_TRIANGLE ([integer](#))

imagick::FILTER_HERMITE ([integer](#))

imagick::FILTER_HANNING ([integer](#))

imagick::FILTER_HAMMING ([integer](#))

imagick::FILTER_BLACKMAN ([integer](#))

imagick::FILTER_GAUSSIAN ([integer](#))

imagick::FILTER_QUADRATIC ([integer](#))

imagick::FILTER_CUBIC ([integer](#))

imagick::FILTER_CATROM ([integer](#))

imagick::FILTER_MITCHELL ([integer](#))

imagick::FILTER_LANCZOS ([integer](#))

imagick::FILTER_BESSEL ([integer](#))

imagick::FILTER_SINC ([integer](#))

Image type constants

imagick::IMGTYPE_UNDEFINED ([integer](#))

imagick::IMGTYPE_BILEVEL ([integer](#))

imagick::IMGTYPE_GRAYSCALE ([integer](#))

imagick::IMGTYPE_GRAYSCALEMATTE ([integer](#))

imagick::IMGTYPE_PALETTE ([integer](#))

imagick::IMGTYPE_PALETTEMATTE ([integer](#))

imagick::IMGTYPE_TRUECOLOR ([integer](#))

imagick::IMGTYPE_TRUECOLORMATTE ([integer](#))

imagick::IMGTYPE_COLORSEPARATION ([integer](#))

imagick::IMGTYPE_COLORSEPARATIONMATTE ([integer](#))

imagick::IMGTYPE_OPTIMIZE ([integer](#))

Resolution constants

imagick::RESOLUTION_UNDEFINED ([integer](#))

imagick::RESOLUTION_PIXELSPERINCH ([integer](#))

imagick::RESOLUTION_PIXELSPERCENTIMETER ([integer](#))

Compression constants

imagick::COMPRESSION_UNDEFINED ([integer](#))

imagick::COMPRESSION_NO ([integer](#))

imagick::COMPRESSION_BZIP ([integer](#))

imagick::COMPRESSION_FAX ([integer](#))

imagick::COMPRESSION_GROUP4 ([integer](#))

imagick::COMPRESSION_JPEG ([integer](#))

imagick::COMPRESSION_JPEG2000 ([integer](#))

imagick::COMPRESSION_LOSSLESSJPEG ([integer](#))

imagick::COMPRESSION_LZW ([integer](#))

imagick::COMPRESSION_RLE ([integer](#))

imagick::COMPRESSION_ZIP ([integer](#))

Paint constants

imagick::PAINT_POINT ([integer](#))

imagick::PAINT_REPLACE ([integer](#))

imagick::PAINT_FLOODFILL ([integer](#))

imagick::PAINT_FILLTOBORDER ([integer](#))

imagick::PAINT_RESET ([integer](#))

Gravity constants

imagick::GRAVITY_NORTHWEST ([integer](#))

imagick::GRAVITY_NORTH ([integer](#))

imagick::GRAVITY_NORTHEAST ([integer](#))

imagick::GRAVITY_WEST ([integer](#))

imagick::GRAVITY_CENTER ([integer](#))

imagick::GRAVITY_EAST ([integer](#))

imagick::GRAVITY_SOUTHWEST ([integer](#))

imagick::GRAVITY_SOUTH ([integer](#))

imagick::GRAVITY_SOUTHEAST ([integer](#))

Stretch constants

imagick::STRETCH_NORMAL ([integer](#))

imagick::STRETCH_ULTRACONDENSED ([integer](#))

imagick::STRETCH_CONDENSED ([integer](#))

imagick::STRETCH_SEMICONDENSED ([integer](#))

imagick::STRETCH_SEMIEXPANDED ([integer](#))

imagick::STRETCH_EXPANDED ([integer](#))

imagick::STRETCH_EXTRAEXPANDED ([integer](#))

imagick::STRETCH_ULTRAEXPANDED ([integer](#))

imagick::STRETCH_ANY ([integer](#))

Align constants

imagick::ALIGN_UNDEFINED ([integer](#))

imagick::ALIGN_LEFT ([integer](#))

imagick::ALIGN_CENTER ([integer](#))

imagick::ALIGN_RIGHT ([integer](#))

Decoration constants

imagick::DECORATION_NO ([integer](#))

imagick::DECORATION_UNDERLINE ([integer](#))

imagick::DECORATION_OVERLINE ([integer](#))

imagick::DECORATION_LINETROUGH ([integer](#))

Noise constants

imagick::NOISE_UNIFORM ([integer](#))

imagick::NOISE_GAUSSIAN ([integer](#))

imagick::NOISE_MULTIPLICATIVEGAUSSIAN ([integer](#))

imagick::NOISE_IMPULSE ([integer](#))

imagick::NOISE_LAPLACIAN ([integer](#))

imagick::NOISE_POISSON ([integer](#))

Channel constants

imagick::CHANNEL_UNDEFINED ([integer](#))

imagick::CHANNEL_RED ([integer](#))

imagick::CHANNEL_GRAY ([integer](#))

imagick::CHANNEL_CYAN ([integer](#))

imagick::CHANNEL_GREEN ([integer](#))

imagick::CHANNEL_MAGENTA ([integer](#))

imagick::CHANNEL_BLUE ([integer](#))

imagemick::CHANNEL_YELLOW ([integer](#))

imagemick::CHANNEL_ALPHA ([integer](#))

imagemick::CHANNEL_OPACITY ([integer](#))

imagemick::CHANNEL_MATTE ([integer](#))

imagemick::CHANNEL_BLACK ([integer](#))

imagemick::CHANNEL_INDEX ([integer](#))

imagemick::CHANNEL_ALL ([integer](#))

Metric constants

imagemick::METRIC_UNDEFINED ([integer](#))

imagemick::METRIC_MEANABSOLUTEERROR ([integer](#))

imagemick::METRIC_MEANSQUAREERROR ([integer](#))

imagemick::METRIC_PEAKABSOLUTEERROR ([integer](#))

imagemick::METRIC_PEAKSIGNALTONOISERATIO ([integer](#))

imagemick::METRIC_ROOTMEANSQUAREDERROR ([integer](#))

Pixel constants

imagemick::PIXEL_CHAR ([integer](#))

imagemick::PIXEL_DOUBLE ([integer](#))

imagemick::PIXEL_FLOAT ([integer](#))

imagemick::PIXEL_INTEGER ([integer](#))

imagemick::PIXEL_LONG ([integer](#))

imagemagick::PIXEL_QUANTUM ([integer](#))

imagemagick::PIXEL_SHORT ([integer](#))

Evaluate Operator constants

imagemagick::EVALUATE_UNDEFINED ([integer](#))

imagemagick::EVALUATE_ADD ([integer](#))

imagemagick::EVALUATE_AND ([integer](#))

imagemagick::EVALUATE_DIVIDE ([integer](#))

imagemagick::EVALUATE_LEFTSHIFT ([integer](#))

imagemagick::EVALUATE_MAX ([integer](#))

imagemagick::EVALUATE_MIN ([integer](#))

imagemagick::EVALUATE_MULTIPLY ([integer](#))

imagemagick::EVALUATE_OR ([integer](#))

imagemagick::EVALUATE_RIGHTSHIFT ([integer](#))

imagemagick::EVALUATE_SET ([integer](#))

imagemagick::EVALUATE_SUBTRACT ([integer](#))

imagemagick::EVALUATE_XOR ([integer](#))

Colorspace constants

imagemagick::COLORSPACE_UNDEFINED ([integer](#))

imagemagick::COLORSPACE_RGB ([integer](#))

imagemagick::COLORSPACE_GRAY ([integer](#))

imagick::COLORSPACE_TRANSPARENT ([integer](#))

imagick::COLORSPACE_OHTA ([integer](#))

imagick::COLORSPACE_LAB ([integer](#))

imagick::COLORSPACE_XYZ ([integer](#))

imagick::COLORSPACE_YCBCR ([integer](#))

imagick::COLORSPACE_YCC ([integer](#))

imagick::COLORSPACE_YIQ ([integer](#))

imagick::COLORSPACE_YPBPR ([integer](#))

imagick::COLORSPACE_YUV ([integer](#))

imagick::COLORSPACE_CMYK ([integer](#))

imagick::COLORSPACE_SRGB ([integer](#))

imagick::COLORSPACE_HSB ([integer](#))

imagick::COLORSPACE_HSL ([integer](#))

imagick::COLORSPACE_HWB ([integer](#))

imagick::COLORSPACE_REC601LUMA ([integer](#))

imagick::COLORSPACE_REC709LUMA ([integer](#))

imagick::COLORSPACE_LOG ([integer](#))

Virtual Pixel Method constants

imagick::VIRTUALPIXELMETHOD_UNDEFINED ([integer](#))

imagick::VIRTUALPIXELMETHOD_BACKGROUND ([integer](#))

imagick::VIRTUALPIXELMETHOD_CONSTANT ([integer](#))

imagick::VIRTUALPIXELMETHOD_EDGE ([integer](#))

imagick::VIRTUALPIXELMETHOD_MIRROR ([integer](#))

imagick::VIRTUALPIXELMETHOD_TILE ([integer](#))

imagick::VIRTUALPIXELMETHOD_TRANSPARENT ([integer](#))

Preview constants

imagick::PREVIEW_UNDEFINED ([integer](#))

imagick::PREVIEW_ROTATE ([integer](#))

imagick::PREVIEW_SHEAR ([integer](#))

imagick::PREVIEW_ROLL ([integer](#))

imagick::PREVIEW_HUE ([integer](#))

imagick::PREVIEW_SATURATION ([integer](#))

imagick::PREVIEW_BRIGHTNESS ([integer](#))

imagick::PREVIEW_GAMMA ([integer](#))

imagick::PREVIEW_SPIFF ([integer](#))

imagick::PREVIEW_DULL ([integer](#))

imagick::PREVIEW_GRAYSCALE ([integer](#))

imagick::PREVIEW_QUANTIZE ([integer](#))

imagick::PREVIEW_DESPECKLE ([integer](#))

imagick::PREVIEW_REDUCE_NOISE ([integer](#))

imagick::PREVIEW_ADD_NOISE ([integer](#))

imagick::PREVIEW_SHARPEN ([integer](#))

imagick::PREVIEW_BLUR ([integer](#))

imagick::PREVIEW_THRESHOLD ([integer](#))

imagick::PREVIEW_EDGE_DETECT ([integer](#))

imagick::PREVIEW_SPREAD ([integer](#))

imagick::PREVIEW_SOLARIZE ([integer](#))

imagick::PREVIEW_SHADE ([integer](#))

imagick::PREVIEW_RAISE ([integer](#))

imagick::PREVIEW_SEGMENT ([integer](#))

imagick::PREVIEW_SWIRL ([integer](#))

imagick::PREVIEW_IMPLODE ([integer](#))

imagick::PREVIEW_WAVE ([integer](#))

imagick::PREVIEW_OIL_PAINT ([integer](#))

imagick::PREVIEW_CHARCOAL_DRAWING ([integer](#))

imagick::PREVIEW_JPEG ([integer](#))

Rendering Intent constants

imagemick::RENDERINGINTENT_UNDEFINED ([integer](#))

imagemick::RENDERINGINTENT_SATURATION ([integer](#))

imagemick::RENDERINGINTENT_PERCEPTUAL ([integer](#))

imagemick::RENDERINGINTENT_ABSOLUTE ([integer](#))

imagemick::RENDERINGINTENT_RELATIVE ([integer](#))

Interlace constants (imagemick::INTERLACE_GIF, imagemick::INTERLACE_JPEG, imagemick::INTERLACE_PNG are only available if Imagemick is compiled against Imagemagick 6.3.5 or newer)

imagemick::INTERLACE_UNDEFINED ([integer](#))

imagemick::INTERLACE_NO ([integer](#))

imagemick::INTERLACE_LINE ([integer](#))

imagemick::INTERLACE_PLANE ([integer](#))

imagemick::INTERLACE_PARTITION ([integer](#))

imagemick::INTERLACE_JPEG ([integer](#))

imagemick::INTERLACE_GIF ([integer](#))

imagemick::INTERLACE_PNG ([integer](#))

Fillrule constants

imagemick::FILLRULE_UNDEFINED ([integer](#))

imagemick::FILLRULE_EVENODD ([integer](#))

imagemick::FILLRULE_NONZERO ([integer](#))

Pathunit constants

imagemick::PATHUNITS_UNDEFINED ([integer](#))

imagemick::PATHUNITS_USERSPACE ([integer](#))

imagemick::PATHUNITS_USERSPACEONUSE ([integer](#))

imagemick::PATHUNITS_OBJECTBOUNDINGBOX ([integer](#))

Linecap constants

imagemick::LINECAP_UNDEFINED ([integer](#))

imagemick::LINECAP_BUTT ([integer](#))

imagemick::LINECAP_ROUND ([integer](#))

imagemick::LINECAP_SQUARE ([integer](#))

Line Join constants

imagemick::LINEJOIN_UNDEFINED ([integer](#))

imagemick::LINEJOIN_MITER ([integer](#))

imagemick::LINEJOIN_ROUND ([integer](#))

imagemick::LINEJOIN_BEVEL ([integer](#))

Resourcetype constants

imagemick::RESOURCE_TYPE_UNDEFINED ([integer](#))

imagemick::RESOURCE_TYPE_AREA ([integer](#))

imagemick::RESOURCE_TYPE_DISK ([integer](#))

imagemick::RESOURCE_TYPE_FILE ([integer](#))

imagemick::RESOURCE_TYPE_MAP ([integer](#))

imagemick::RESOURCE_TYPE_MEMORY ([integer](#))

Layer Method constants (available if compiled against ImageMagick 6.3.3 or later)

imagemick::LAYERMETHOD_UNDEFINED ([integer](#))

imagemick::LAYERMETHOD_COALESCE ([integer](#))

imagemick::LAYERMETHOD_COMPAREANY ([integer](#))

imagemick::LAYERMETHOD_COMPARECLEAR ([integer](#))

imagemick::LAYERMETHOD_COMPAREOVERLAY ([integer](#))

imagemick::LAYERMETHOD_DISPOSE ([integer](#))

imagemick::LAYERMETHOD_OPTIMIZE ([integer](#))

imagemick::LAYERMETHOD_OPTIMIZEIMAGE ([integer](#))

imagemick::LAYERMETHOD_OPTIMIZEPLUS ([integer](#))

imagemick::LAYERMETHOD_OPTIMIZEPLUS ([integer](#))

imagemick::LAYERMETHOD_REMOVEDUPS ([integer](#))

imagemick::LAYERMETHOD_REMOVEZERO ([integer](#))

imagemick::LAYERMETHOD_COMPOSITE ([integer](#))

Orientation constants (available if compiled against ImageMagick 6.3.4 or later)

imagemick::ORIENTATION_UNDEFINED ([integer](#))

imagemick::ORIENTATION_TOPLEFT ([integer](#))

imagemick::ORIENTATION_TOPRIGHT ([integer](#))

imagemick::ORIENTATION_BOTTOMRIGHT ([integer](#))

imagemick::ORIENTATION_BOTTOMLEFT ([integer](#))

imagemick::ORIENTATION_LEFTTOP ([integer](#))

imagick::ORIENTATION_RIGHTTOP ([integer](#))

imagick::ORIENTATION_RIGHTBOTTOM ([integer](#))

imagick::ORIENTATION_LEFTBOTTOM ([integer](#))

Distortion constants (available if compiled against ImageMagick 6.3.6 or later)

imagick::DISTORTION_UNDEFINED ([integer](#))

imagick::DISTORTION_AFFINE ([integer](#))

imagick::DISTORTION_AFFINEPROJECTION ([integer](#))

imagick::DISTORTION_ARC ([integer](#))

imagick::DISTORTION_BILINEAR ([integer](#))

imagick::DISTORTION_PERSPECTIVE ([integer](#))

imagick::DISTORTION_PERSPECTIVEPROJECTION ([integer](#))

imagick::DISTORTION_SCALEROTATETRANSLATE ([integer](#))

Examples

Examples

Imagick makes image manipulation in PHP extremely easy through an OO interface. Here is a quick example on how to make a thumbnail:

Example #1 - Creating a thumbnail in Imagick

```
<?php

header('Content-type: image/jpeg');

$image = new Imagick('image.jpg');

// If 0 is provided as a width or height parameter,
// aspect ratio is maintained
$image->thumbnailImage(100, 0);

echo $image;

?>
```

Using SPL and other OO features supported in Imagick, it can be simple to resize all files in a directory (useful for batch resizing large digital camera images to be web viewable). Here we use `resize`, as we might want to retain certain meta-data:

Example #2 - Make a thumbnail of all JPG files in a directory

```
<?php

$images = new Imagick(glob('images/*.JPG'));

foreach($images as $image) {

    // Providing 0 forces thumbnailImage to maintain aspect ratio
    $image->thumbnailImage(1024,0);

}

$images->writeImages();

?>
```

This is an example of creating a reflection of an image. The reflection is created by flipping the image and overlaying a gradient on it. Then both, the original image and the reflection is overlayed on a canvas.

Example #3 - Creating a reflection of an image

```
<?php
/* Read the image */
$im = new Imagick("test.png");

/* Thumbnail the image */
$im->thumbnailImage(200, null);

/* Create a border for the image */
$im->borderImage(new ImagickPixel("white"), 5, 5);

/* Clone the image and flip it */
$reflection = $im->clone();
$reflection->flipImage();

/* Create gradient. It will be overlayd on the reflection */
$gradient = new Imagick();

/* Gradient needs to be large enough for the image and the borders */
$gradient->newPseudoImage($reflection->getImageWidth() + 10,
$reflection->getImageHeight() + 10, "gradient:transparent-black");

/* Composite the gradient on the reflection */
$reflection->compositeImage($gradient, imagick::COMPOSITE_OVER, 0, 0);

/* Add some opacity. Requires ImageMagick 6.2.9 or later */
$reflection->setImageOpacity( 0.3 );

/* Create an empty canvas */
$canvas = new Imagick();

/* Canvas needs to be large enough to hold the both images */
$width = $im->getImageWidth() + 40;
$height = ($im->getImageHeight() * 2) + 30;
$canvas->newImage($width, $height, new ImagickPixel("black"));
$canvas->setImageFormat("png");

/* Composite the original image and the reflection on the canvas */
$canvas->compositeImage($im, imagick::COMPOSITE_OVER, 20, 10);
$canvas->compositeImage($reflection, imagick::COMPOSITE_OVER, 20,
$im->getImageHeight() + 10);

/* Output the image*/
header("Content-Type: image/png");
echo $canvas;
?>
```

This example illustrates how to use fill patterns during drawing.

Example #4 - Filling text with gradient

```
<?php

/* Create a new imagick object */
$im = new Imagick();
```

```
/* Create new image. This will be used as fill pattern */
$im->newPseudoImage(50, 50, "gradient:red-black");

/* Create imagickdraw object */
$draw = new ImagickDraw();

/* Start a new pattern called "gradient" */
$draw->pushPattern('gradient', 0, 0, 50, 50);

/* Composite the gradient on the pattern */
$draw->composite(Imagick::COMPOSITE_OVER, 0, 0, 50, 50, $im);

/* Close the pattern */
$draw->popPattern();

/* Use the pattern called "gradient" as the fill */
$draw->setFillPatternURL('#gradient');

/* Set font size to 52 */
$draw->setFontSize(52);

/* Annotate some text */
$draw->annotation(20, 50, "Hello World!");

/* Create a new canvas object and a white image */
$canvas = new Imagick();
$canvas->newImage(350, 70, "white");

/* Draw the ImagickDraw on to the canvas */
$canvas->drawImage($draw);

/* 1px black border around the image */
$canvas->borderImage('black', 1, 1);

/* Set the format to PNG */
$canvas->setImageFormat('png');

/* Output the image */
header("Content-Type: image/png");
echo $canvas;
?>
```

The Imagick class

Class synopsis

| |
|----------------|
| Imagick |
|----------------|

Imagick implements Iterator, Traversable {

bool **Imagick::adaptiveBlurImage** (float \$radius, float \$sigma [, int \$channel])

bool **Imagick::adaptiveResizeImage** (int \$columns, int \$rows [, bool \$fit])

bool **Imagick::adaptiveSharpenImage** (float \$radius, float \$sigma [, int \$channel])

bool **Imagick::adaptiveThresholdImage** (int \$width, int \$height, int \$offset)

bool **Imagick::addImage** ([Imagick](#) \$source)

bool **Imagick::addNoiseImage** (int \$noise_type [, int \$channel])

bool **Imagick::affineTransformImage** ([ImagickDraw](#) \$matrix)

bool **Imagick::annotateImage** ([ImagickDraw](#) \$draw_settings, float \$x, float \$y, float \$angle, string \$text)

Imagick **Imagick::appendImages** (bool \$stack)

Imagick **Imagick::averageImages** (void)

bool **Imagick::blackThresholdImage** ([mixed](#) \$threshold)

bool **Imagick::blurImage** (float \$radius, float \$sigma [, int \$channel])

bool **Imagick::borderImage** ([mixed](#) \$bordercolor, int \$width, int \$height)

bool **Imagick::charcoalImage** (float \$radius, float \$sigma)

bool **Imagick::chopImage** (int \$width, int \$height, int \$x, int \$y)

bool **Imagick::clear** (void)

bool **Imagick::clipImage** (void)

bool **Imagick::clipPathImage** (string \$pathname, bool \$inside)

Imagick Imagick::clone (void)

bool Imagick::clutImage ([Imagick](#) \$lookup_table [, int \$channel])

Imagick Imagick::coalesceImages (void)

bool Imagick::colorFloodfillImage ([mixed](#) \$fill, float \$fuzz, [mixed](#) \$bordercolor, int \$x, int \$y)

bool Imagick::colorizeImage ([mixed](#) \$colorize, [mixed](#) \$opacity)

Imagick Imagick::combineImages (int \$channelType)

bool Imagick::commentImage (string \$comment)

Imagick Imagick::compareImageChannels ([Imagick](#) \$image, int \$channelType, int \$metricType)

Imagick Imagick::compareImageLayers (int \$method)

array Imagick::compareImages ([Imagick](#) \$compare, int \$metric)

bool Imagick::compositeImage ([Imagick](#) \$composite_object, int \$composite, int \$x, int \$y [, int \$channel])

Imagick Imagick::__construct ([[mixed](#) \$files])

bool Imagick::contrastImage (bool \$sharpen)

bool Imagick::contrastStretchImage (float \$black_point, float \$white_point [, int \$channel])

bool Imagick::convolveImage (array \$kernel [, int \$channel])

bool Imagick::cropImage (int \$width, int \$height, int \$x, int \$y)

bool Imagick::cropThumbnailImage (int \$width, int \$height)

Imagick Imagick::current (void)

bool Imagick::cycleColormapImage (int \$displace)

bool Imagick::deconstructImages (void)

bool Imagick::despeckleImage (void)

bool Imagick::destroy (void)

bool Imagick::displayImage (string \$servername)

bool Imagick::displayImages (string \$servername)

bool Imagick::distortImage (int \$method, array \$arguments, bool \$bestfit)

bool Imagick::drawImage ([ImagickDraw](#) \$draw)

bool Imagick::edgelImage (float \$radius)

bool Imagick::embossImage (float \$radius, float \$sigma)

bool Imagick::enhanceImage (void)

bool Imagick::equalizeImage (void)

bool Imagick::evaluateImage (int \$op, float \$constant [, int \$channel])

Imagick Imagick::flattenImages (void)

bool Imagick::fliplImage (void)

bool Imagick::flopImage (void)

bool Imagick::frameImage ([mixed](#) \$matte_color, int \$width, int \$height, int \$inner_bevel, int \$outer_bevel)

Imagick Imagick::fxImage (string \$expression [, int \$channel])

bool Imagick::gammaImage (float \$gamma [, int \$channel])

bool Imagick::gaussianBlurImage (float \$radius, float \$sigma [, int \$channel])

int Imagick::getCompression (void)

int Imagick::getCompressionQuality (void)

string Imagick::getCopyright (void)

string Imagick::getFilename (void)

string Imagick::getFormat (void)

string Imagick::getHomeURL (void)

Imagick Imagick::getImage (void)

ImagickPixel Imagick::getImageBackgroundColor (void)

string Imagick::getImageBlob (void)

ImagickPixel Imagick::getImageBluePrimary (float \$x, float \$y)

ImagickPixel Imagick::getImageBorderColor (void)

int Imagick::getImageChannelDepth (int \$channelType)

float Imagick::getImageChannelDistortion ([Imagick](#) \$reference, int \$channel, int \$metric)

array Imagick::getImageChannelExtrema (int \$channel)

array Imagick::getImageChannelMean (int \$channel)

array Imagick::getImageChannelStatistics (void)

ImagickPixel Imagick::getImageColormapColor (int \$index)

int Imagick::getImageColors (void)

int Imagick::getImageColorspace (void)

int Imagick::getImageCompose (void)

int Imagick::getImageDelay (void)

int Imagick::getImageDepth (void)

int Imagick::getImageDispose (void)

float Imagick::getImageDistortion ([MagickWand](#) \$reference, int \$metric)

array Imagick::getImageExtrema (void)

string Imagick::getImageFilename (void)

string Imagick::getImageFormat (void)

float Imagick::getImageGamma (void)

array Imagick::getImageGeometry (void)

array Imagick::getImageGreenPrimary (void)

int Imagick::getImageHeight (void)

array Imagick::getImageHistogram (void)

int Imagick::getImageIndex (void)

int Imagick::getImageInterlaceScheme (void)

int Imagick::getImageInterpolateMethod (void)

int Imagick::getImageIterations (void)

int Imagick::getImageLength (void)

string Imagick::getImageMagickLicense (void)

int Imagick::getImageMatte (void)

ImagickPixel Imagick::getImageMatteColor (void)

int Imagick::getImageOrientation (void)

array Imagick::getImagePage (void)

ImagickPixel Imagick::getImagePixelColor (int \$x, int \$y)

string Imagick::getImageProfile (string \$name)

array Imagick::getImageProfiles ([string \$pattern [, bool \$only_names]])

array Imagick::getImageProperties ([string \$pattern [, bool \$only_names]])

string Imagick::getImageProperty (string \$name)

array Imagick::getImageRedPrimary (void)

Imagick Imagick::getImageRegion (int \$width, int \$height, int \$x, int \$y)

int Imagick::getImageRenderingIntent (void)

array Imagick::getImageResolution (void)

int Imagick::getImageScene (void)

string Imagick::getImageSignature (void)

int Imagick::getImageSize (void)

int Imagick::getImageTicksPerSecond (void)

float Imagick::getImageTotalInkDensity (void)

int Imagick::getImageType (void)

int Imagick::getImageUnits (void)

int Imagick::getImageVirtualPixelMethod (void)

array Imagick::getImageWhitePoint (void)

int Imagick::getImageWidth (void)

int Imagick::getImageInterlaceScheme (void)

int Imagick::getImageIteratorIndex (void)

int Imagick::getImageNumberImages (void)

string Imagick::getOption (string \$key)

string Imagick::getPackageName (void)

array Imagick::getPage (void)

ImagickPixelIterator Imagick::getPixelIterator (void)

ImagickPixelIterator Imagick::getPixelRegionIterator (int \$x, int \$y, int \$columns, int \$rows)

array Imagick::getQuantumDepth (void)

array Imagick::getQuantumRange (void)

string Imagick::getReleaseDate (void)

int Imagick::getResource (int \$type)

int Imagick::getResourceLimit (int \$type)

array Imagick::getSamplingFactors (void)

array Imagick::getSize (void)

int Imagick::getSizeOffset (void)

array Imagick::getVersion (void)

bool Imagick::hasNextImage (void)

bool Imagick::hasPreviousImage (void)

array Imagick::identifyImage ([bool \$appendRawOutput])

bool Imagick::implodeImage (float \$radius)

bool Imagick::labelImage (string \$label)

bool Imagick::levelImage (float \$blackPoint, float \$gamma, float \$whitePoint [, int \$channel])

bool Imagick::linearStretchImage (float \$blackPoint, float \$whitePoint)

bool Imagick::magnifyImage (void)

bool Imagick::mapImage ([Imagick](#) \$map, bool \$dither)

bool Imagick::matteFloodfillImage (float \$alpha, float \$fuzz, [mixed](#) \$bordercolor, int \$x, int \$y)

bool Imagick::medianFilterImage (float \$radius)

bool Magick::minifyImage (void)

bool Magick::modulateImage (float \$brightness, float \$saturation, float \$hue)

Magick Magick::montageImage ([ImagickDraw](#) \$draw, string \$tile_geometry, string \$thumbnail_geometry, int \$mode, string \$frame)

Magick Magick::morphImages (int \$number_frames)

Magick Magick::mosaicImages (void)

bool Magick::motionBlurImage (float \$radius, float \$sigma, float \$angle)

bool Magick::negateImage (bool \$gray [, int \$channel])

bool Magick::newImage (int \$cols, int \$rows, [mixed](#) \$background [, string \$format])

bool Magick::newPseudoImage (int \$columns, int \$rows, string \$pseudoString)

bool Magick::nextImage (void)

bool Magick::normalizeImage ([int \$channel])

bool Magick::oilPaintImage (float \$radius)

bool Magick::optimizeImageLayers (void)

bool Magick::paintFloodfillImage ([mixed](#) \$fill, float \$fuzz, [mixed](#) \$bordercolor, int \$x, int \$y)

bool Magick::paintOpaqueImage ([mixed](#) \$target, [mixed](#) \$fill, float \$fuzz [, int \$channel])

bool Magick::paintTransparentImage ([mixed](#) \$target, float \$alpha, float \$fuzz)

bool Magick::pingImage (string \$filename)

bool Magick::pingImageBlob (string \$image)

bool Magick::pingImageFile (resource \$filehandle [, string \$fileName])

bool Magick::polaroidImage ([ImagickDraw](#) \$properties, float \$angle)

bool Magick::posterizeImage (int \$levels, bool \$dither)

bool Magick::previewImages (int \$preview)

bool Magick::previousImage (void)

bool Magick::profileImage (string \$name, string \$profile)

```
bool Imagick::quantizeImage ( int $numberColors, int $colorspace, int $treedepth,
bool $dither, bool $measureError )

bool Imagick::quantizeImages ( int $numberColors, int $colorspace, int $treedepth,
bool $dither, bool $measureError )

array Imagick::queryFontMetrics ( ImagickDraw $properties, string $text [, bool $
multiline ] )

array Imagick::queryFonts ( [ string $pattern ] )

array Imagick::queryFormats ( [ string $pattern ] )

bool Imagick::radialBlurImage ( float $angle [, int $channel ] )

bool Imagick::raiseImage ( int $width, int $height, int $x, int $y, bool $raise )

bool Imagick::randomThresholdImage ( float $low, float $high [, int $channel ] )

bool Imagick::readImage ( string $filename )

bool Imagick::readImageBlob ( string $image [, string $filename ] )

bool Imagick::readImageFile ( resource $filehandle [, string $fileName ] )

bool Imagick::reduceNoiseImage ( float $radius )

bool Imagick::removeImage ( void )

string Imagick::removeImageProfile ( string $name )

bool Imagick::render ( void )

bool Imagick::resampleImage ( float $x_resolution, float $y_resolution, int $
filter, float $blur )

bool Imagick::resizeImage ( int $columns, int $rows, int $filter, float $blur [, bool $
fit ] )

bool Imagick::rollImage ( int $x, int $y )

bool Imagick::rotateImage ( mixed $background, float $degrees )

bool Imagick::roundCorners ( float $x_rounding, float $y_rounding [, float $
stroke_width [, float $displace [, float $size_correction ] ] ] )

bool Imagick::sampleImage ( int $columns, int $rows )

bool Imagick::scaleImage ( int $cols, int $rows [, bool $fit ] )

bool Imagick::separateImageChannel ( int $channel )
```

bool Magick::sepiaToneImage (float \$threshold)

bool Magick::setBackgroundColor ([mixed](#) \$background)

bool Magick::setCompression (int \$compression)

bool Magick::setCompressionQuality (int \$quality)

bool Magick::setFilename (string \$filename)

bool Magick::setFirstIterator (void)

bool Magick::setFormat (string \$format)

bool Magick::setImage ([Imagick](#) \$replace)

bool Magick::setImageBackgroundColor ([mixed](#) \$background)

bool Magick::setImageBias (float \$bias)

bool Magick::setImageBluePrimary (float \$x, float \$y)

bool Magick::setImageBorderColor ([mixed](#) \$border)

bool Magick::setImageChannelDepth (int \$channel, int \$depth)

bool Magick::setImageColormapColor (int \$index, [ImagickPixel](#) \$color)

bool Magick::setImageColorspace (int \$colorspace)

bool Magick::setImageCompose (int \$compose)

bool Magick::setImageCompression (int \$compression)

bool Magick::setImageDelay (int \$delay)

bool Magick::setImageDepth (int \$depth)

bool Magick::setImageDispose (int \$dispose)

bool Magick::setImageExtent (int \$columns, int \$rows)

bool Magick::setImageFilename (string \$filename)

bool Magick::setImageFormat (string \$format)

bool Magick::setImageGamma (float \$gamma)

bool Magick::setImageGreenPrimary (float \$x, float \$y)

bool Magick::setImageIndex (int \$index)

bool Magick::setImageInterlaceScheme (int \$interlace_scheme)

bool Magick::setImageInterpolateMethod (int \$method)

bool Magick::setImageIterations (int \$iterations)

bool Magick::setImageMatte (bool \$matte)

bool Magick::setImageMatteColor ([mixed](#) \$matte)

bool Magick::setImageOpacity (float \$opacity)

bool Magick::setImageOrientation (int \$orientation)

bool Magick::setImagePage (int \$width, int \$height, int \$x, int \$y)

bool Magick::setImageProfile (string \$name, string \$profile)

bool Magick::setImageProperty (string \$name, string \$value)

bool Magick::setImageRedPrimary (float \$x, float \$y)

bool Magick::setImageRenderingIntent (int \$rendering_intent)

bool Magick::setImageResolution (float \$x_resolution, float \$y_resolution)

bool Magick::setImageScene (int \$scene)

bool Magick::setImageTicksPerSecond (int \$ticks_per-second)

bool Magick::setImageType (int \$image_type)

bool Magick::setImageUnits (int \$units)

bool Magick::setImageVirtualPixelMethod (int \$method)

bool Magick::setImageWhitePoint (float \$x, float \$y)

bool Magick::setInterlaceScheme (int \$interlace_scheme)

bool Magick::setIteratorIndex (int \$index)

bool Magick::setLastIterator (void)

bool Magick::setOption (string \$key, string \$value)

bool Magick::setPage (int \$width, int \$height, int \$x, int \$y)

bool Magick::setResolution (float \$x_resolution, float \$y_resolution)

bool Magick::setResourceLimit (int \$type, int \$limit)

bool Imagick::setSamplingFactors (array \$factors)

bool Imagick::setSize (int \$columns, int \$rows)

bool Imagick::setSizeOffset (int \$columns, int \$rows, int \$offset)

bool Imagick::setType (int \$image_type)

bool Imagick::shadelImage (bool \$gray, float \$azimuth, float \$elevation)

bool Imagick::shadowImage (float \$opacity, float \$sigma, int \$x, int \$y)

bool Imagick::sharpenImage (float \$radius, float \$sigma [, int \$channel])

bool Imagick::shaveImage (int \$columns, int \$rows)

bool Imagick::shearImage ([mixed](#) \$background, float \$x_shear, float \$y_shear)

bool Imagick::sigmoidalContrastImage (bool \$sharpen, float \$alpha, float \$beta [, int \$channel])

bool Imagick::sketchImage (float \$radius, float \$sigma, float \$angle)

bool Imagick::solarizeImage (int \$threshold)

bool Imagick::spliceImage (int \$width, int \$height, int \$x, int \$y)

bool Imagick::spreadImage (float \$radius)

Imagick Imagick::steganolImage ([Imagick](#) \$watermark_wand, int \$offset)

bool Imagick::stereolImage ([Imagick](#) \$offset_wand)

bool Imagick::striplImage (void)

bool Imagick::swirlImage (float \$degrees)

bool Imagick::textureImage ([Imagick](#) \$texture_wand)

bool Imagick::thresholdImage (float \$threshold [, int \$channel])

bool Imagick::thumbnailImage (int \$columns, int \$rows [, bool \$fit])

bool Imagick::tintImage ([mixed](#) \$tint, [mixed](#) \$opacity)

Imagick Imagick::transformImage (string \$crop, string \$geometry)

bool Imagick::transverseImage (void)

bool Imagick::trimImage (float \$fuzz)

bool Imagick::uniqueImageColors (void)

```

bool Imagick::unsharpMaskImage ( float $radius, float $sigma, float $amount, float
$threshold [, int $channel ] )

bool Imagick::valid ( void )

bool Imagick::vignetteImage ( float $blackPoint, float $whitePoint, int $x, int $y )

bool Imagick::wavelImage ( float $amplitude, float $length )

bool Imagick::whiteThresholdImage ( mixed $threshold )

bool Imagick::writeImage ( [ string $filename ] )

bool Imagick::writeImages ( string $filename, bool $adjoin )
}

```

Image methods and global methods

The Imagick class has the ability to hold and operate on multiple images simultaneously. This is achieved through an internal stack. There is always an internal pointer that points at the current image. Some functions operate on all images in the Imagick class, but most operate only on the current image in the internal stack. As a convention, method names can contain the word Image to denote they affect only the current image in the stack.

Class Methods

Because there are so many methods, here is a handy list of methods, somewhat reduced to their general purpose:

Class methods by purpose

| Image effects | Get methods | Set methods | Read/write images | Other |
|--|--|--|------------------------------|------------------------------|
| adaptiveBlurImage | getCompression | setBackgroundC olor | __construct | clear |
| adaptiveResizeI mage | getFilename | setCompression Quality | addImage | clone |
| adaptiveSharpen Image | getFormat | setCompression | appendImages | current |
| adaptiveTreshold Image | getImageBackgr oundColor | setFilename | getFilename | destroy |
| addNoiseImage | getImageBlob | setFormat | getFormat | getCopyright |
| | | | | |

| | | | | |
|--------------------------------------|--|---|----------------------------------|----------------------------------|
| affinetransformimage | getImageBluePrimary | setImageBackgroundColor | getImageFilename | getHomeURL |
| annotatImage | getImageBorderColor | setFirstIterator | getImageFormat | commentImage |
| averagImages | getImageChannelDepth | setImageBias | getImage | getNumberImages |
| blackThresholdImage | getImageChannelIDistortion | setImageBluePrimary | setImageFilename | getReleaseDate |
| blurImage | getImageChannelExtrema | setImageBorderColor | setImageFormat | getVersion |
| borderImage | getImageChannelMean | setImageChannelIDepth | readImageFile | hasNextImage |
| charcoalImage | getImageChannelStatistics | setImageColorMapColor | readImage | hasPreviousImage |
| chopImage | getImageColorMapColor | setImageColorSpace | writeImages | labelImage |
| clipImage | getImageColorspace | setImageComposite | writeImage | newImage |
| clipPathImage | getImageColors | setImageCompression | | newPseudoImage |
| coalescelImages | getImageComposite | setImageDelay | | nextImage |
| colorFloodFillImage | getImageDelay | setImageDepth | | pingImageBlob |
| colorizeImage | getImageDepth | setImageDispose | | pingImageFile |
| combinelImages | getImageDispose | setImageDispose | | pingImage |
| compareImageChannels | getImageDistortion | setImageExtent | | previousImage |
| compareImageLayers | getImageExtrema | setImageFilename | | profileImage |
| compositImage | getImageFilename | setImageFormat | | queryFormats |
| contrastImage | getImageFormat | setImageGamma | | removeImagePro |

| | | | | |
|---------------------------------------|---|--|--|----------------------------------|
| | | | | file |
| constrastStretchImage | getImageGamma | setImageGreenPrimary | | removeImage |
| convolveImage | getImageGeometry | setImageIndex | | setFirstIterator |
| cropImage | getImageGreenPrimary | setImageInterpolateMethod | | setImageIndex |
| cycleColormapImage | getImageHeight | setImageIterations | | valid |
| deconstructImages | getImageHistogram | setImageMatteColor | | |
| drawImage | getImageIndex | setImageMatte | | |
| edgeImage | getImageInterlaceScheme | setImagePage | | |
| embossImage | getImageInterpolateMethod | setImageProfile | | |
| enhanceImage | setImageIterations | setImageProperty | | |
| equalizeImage | getImageMatteColor | setImageRedPrimary | | |
| evaluateImage | getImageMatte | setImageRenderingIntent | | |
| flattenImages | getImagePage | setImageResolution | | |
| flipImage | getImagePixelColor | setImageScene | | |
| flopImage | getImageProfile | setImageTicksPerSecond | | |
| imageImage | getImageProperty | setImageType | | |
| fxImage | getImageRedPrimary | setImageUnits | | |
| gammalImage | getImageRegion | setImageVirtualPixelMethod | | |

| | | | | |
|---------------------------------------|--|------------------------------------|--|--|
| gaussianBlurImage | getImageRenderingIntent | setImageWhitepoint | | |
| implodeImage | getImageResolution | setInterlaceScheme | | |
| levelImage | getImageScene | setOption | | |
| linearStretchImage | getImageSignature | setPage | | |
| magnifyImage | getImageTicksPerSecond | setResolution | | |
| matteFloodFillImage | getImageTotalInkDensity | setResourceLimit | | |
| medianFilterImage | getImageType | setSamplingFactors | | |
| minifyImage | getImageUnits | setSizeOffset | | |
| modulateImage | getImageVirtualPixelFormatMethod | setSize | | |
| montageImage | getImageWhitepoint | setType | | |
| morphImages | getImageWidth | | | |
| mosaicImages | getImage | | | |
| motionBlurImage | getInterlaceScheme | | | |
| negateImage | getNumberImages | | | |
| normalizeImage | getOption | | | |
| oilPaintImage | getPackageName | | | |
| optimizeImageLayers | getPage | | | |
| paintOpaqueImage | getPixelIterator | | | |
| paintTransparentImage | getPixelRegionIterator | | | |

| | | | | |
|--|------------------------------------|--|--|--|
| posterizeImage | getQuantumDepth | | | |
| radialBlurImage | getQuantumRange | | | |
| raiseImage | getResourceLimit | | | |
| randomThresholdImage | getResource | | | |
| reduceNoiseImage | getSamplingFactors | | | |
| render | getSizeOffset | | | |
| resampleImage | getSize | | | |
| resizeImage | identifyImage | | | |
| rollImage | getImageSize | | | |
| rotateImage | | | | |
| sampleImage | | | | |
| scaleImage | | | | |
| separateImageChannel | | | | |
| sepiaToneImage | | | | |
| shadeImage | | | | |
| shadowImage | | | | |
| sharpenImage | | | | |
| shaveImage | | | | |
| shearImage | | | | |
| sigmoidalContrastImage | | | | |
| sketchImage | | | | |
| solarizeImage | | | | |
| spliceImage | | | | |

| | | | | |
|-------------------------------------|--|--|--|--|
| spreadImage | | | | |
| steganolImage | | | | |
| stereoImage | | | | |
| stripImage | | | | |
| swirlImage | | | | |
| textureImage | | | | |
| thresholdImage | | | | |
| thumbnailImage | | | | |
| tintImage | | | | |
| transverseImage | | | | |
| trimImage | | | | |
| uniqueImageColors | | | | |
| unsharpMaskImage | | | | |
| vignetteImage | | | | |
| waveImage | | | | |
| whiteThresholdImage | | | | |

Imagick::adaptiveBlurImage

Imagick::adaptiveBlurImage -- Adds adaptive blur filter to image

Description

bool **Imagick::adaptiveBlurImage** (float \$radius, float \$sigma [, int \$channel])

Adds an adaptive blur filter to image. The intensity of an adaptive blur depends is dramatically decreased at edge of the image, whereas a standard blur is uniform across the image.

Parameters

radius

The radius of the Gaussian, in pixels, not counting the center pixel. Provide a value of 0 and the radius will be chosen automatically.

sigma

The standard deviation of the Gaussian, in pixels.

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Examples

Example #5 - Using [Imagick::adaptiveBlurImage\(\)](#):

Adaptively blur an image, then display to the browser.

```
<?php
header('Content-type: image/jpeg');

$image = new Imagick('test.jpg');
```



```
$image->adaptiveBlurImage(5,3);  
echo $image;  
  
?>
```

See Also

- [Imagick::blurImage\(\)](#)
- [Imagick::motionBlurImage\(\)](#)
- [Imagick::radialBlurImage\(\)](#)

Imagick::adaptiveResizeImage

Imagick::adaptiveResizeImage -- Adaptively resize image with data dependent triangulation

Description

bool **Imagick::adaptiveResizeImage** (int \$columns, int \$rows [, bool \$fit])

Adaptively resize image with data-dependent triangulation. Avoids blurring across sharp color changes. Most useful when used to shrink images slightly to a slightly smaller "web size"; may not look good when a full-sized image is adaptively resized to a thumbnail.

Parameters

columns

The number of columns in the scaled image.

rows

The number of rows in the scaled image.

fit

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|--|
| 2.1.0 | Added optional fit parameter. This method now supports proportional scaling. Pass zero as either parameter for proportional scaling. |

Examples

Example #6 - Using [Imagick::adaptiveResizeImage\(\)](#)

Resize an image to a standard size for the web. This method works best when resizing to a size only slightly smaller than the previous image size.

```
<?php
header('Content-type: image/jpeg');

$image = new Imagick('image.jpg');
$image->adaptiveResizeImage(1024,768);

echo $image;
?>
```

See Also

- [Imagick::chopImage\(\)](#)
- [Imagick::cropImage\(\)](#)
- [Imagick::magnifyImage\(\)](#)
- [Imagick::minifyImage\(\)](#)
- [Imagick::resizeImage\(\)](#)
- [Imagick::scaleImage\(\)](#)
- [Imagick::shaveImage\(\)](#)
- [Imagick::thumbnailImage\(\)](#)
- [Imagick::trimImage\(\)](#)

Imagick::adaptiveSharpenImage

Imagick::adaptiveSharpenImage -- Adaptively sharpen the image

Description

bool **Imagick::adaptiveSharpenImage** (float *\$radius*, float *\$sigma* [, int *\$channel*])

Warning

This function is currently not documented; only its argument list is available.

Adaptively sharpen the image by sharpening more intensely near image edges and less intensely far from edges.

Parameters

radius

The radius of the Gaussian, in pixels, not counting the center pixel. Use 0 for auto-select.

sigma

The standard deviation of the Gaussian, in pixels.

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Examples

Example #7 - A [Imagick::adaptiveSharpenImage\(\)](#) example

Adaptively sharpen the image with radius 2 and sigma 1.

```
<?php
try {
    $image = new Imagick('image.png');
    $image->adaptiveSharpenImage(2,1);
}
```

```
} catch(ImagickException $e) {  
    echo 'Error: ' , $e->getMessage();  
    die();  
}  
header('Content-type: image/png');  
echo $image;  
?>
```

See Also

- [Imagick::sharpenImage\(\)](#)

Imagick::adaptiveThresholdImage

Imagick::adaptiveThresholdImage -- Selects a threshold for each pixel based on a range of intensity

Description

bool **Imagick::adaptiveThresholdImage** (int \$width, int \$height, int \$offset)

Selects an individual threshold for each pixel based on the range of intensity values in its local neighborhood. This allows for thresholding of an image whose global intensity histogram doesn't contain distinctive peaks.

Parameters

width

Width of the local neighborhood.

height

Height of the local neighborhood.

offset

The mean offset

Return Values

Returns **TRUE** on success.

Imagick::addImage

Imagick::addImage -- Adds new image to Imagick object image list

Description

bool **Imagick::addImage** ([Imagick](#) \$source)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Adds new image to Imagick object from the current position of the source object. After the operation iterator position is moved at the end of the list.

Parameters

source

The source Imagick object

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::addNoiseImage

Imagick::addNoiseImage -- Adds random noise to the image

Description

bool **Imagick::addNoiseImage** (int *\$noise_type* [, int *\$channel*])

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Adds random noise to the image.

Parameters

noise_type

The type of the noise. Refer to this list of [noise constants](#).

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Imagick::affineTransformImage

Imagick::affineTransformImage -- Transforms an image

Description

bool **Imagick::affineTransformImage** ([ImagickDraw](#) \$matrix)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Transforms an image as dictated by the affine matrix.

Parameters

matrix

The affine matrix

Return Values

Returns **TRUE** on success.

Imagick::annotateImage

Imagick::annotateImage -- Annotates an image with text

Description

bool **Imagick::annotateImage** ([ImagickDraw](#) \$draw_settings, float \$x, float \$y, float \$angle, string \$text)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Annotates an image with text.

Parameters

draw_settings

The ImagickDraw object that contains settings for drawing the text

x

Horizontal offset in pixels to the left of text

y

Vertical offset in pixels to the baseline of text

angle

The angle at which to write the text

text

The string to draw

Return Values

Returns **TRUE** on success.

Examples

| Example #8 - Using Imagick::annotateImage() : |
|---|
| Annotate text on an empty image |
| <?php |

```
/* Create some objects */
$image = new Imagick();
$draw = new ImagickDraw();
$pixel = new ImagickPixel( 'gray' );

/* New image */
$image->newImage(800, 75, $pixel);

/* Black text */
$pixel->setColor('black');

/* Font properties */
$draw->setFont('Bookman-DemiItalic');
$draw->setFontSize( 30 );

/* Create text */
$image->annotateImage($draw, 10, 45, 0, 'The quick brown fox jumps over the
lazy dog');

/* Give image a format */
$image->setImageFormat('png');

/* Output the image with headers */
header('Content-type: image/png');
echo $image;

?>
```

See Also

- [ImagickDraw::annotation\(\)](#)
- [ImagickDraw::setFont\(\)](#)

Imagick::appendImages

Imagick::appendImages -- Append a set of images

Description

[Imagick](#) **Imagick::appendImages** (bool *\$stack*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Append a set of images.

Parameters

stack

The direction of the stack (top to bottom or bottom to top)

Return Values

Returns Imagick instance on success, throws ImagickException on failure.

Imagick::averagelImages

Imagick::averagelImages -- Average a set of images

Description

[Imagick](#) Imagick::averagelImages (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Average a set of images.

Return Values

Returns a new Imagick object on success, throws ImagickException on failure.

Imagick::blackThresholdImage

Imagick::blackThresholdImage -- Forces all pixels below the threshold into black

Description

bool **Imagick::blackThresholdImage** (*mixed* \$threshold)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Is like Imagick::thresholdImage() but forces all pixels below the threshold into black while leaving all pixels above the threshold unchanged.

Parameters

threshold

The threshold below which everything turns black

Return Values

Returns **TRUE** on success.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as a parameter. Previous versions allow only an ImagickPixel object. |

Imagick::blurImage

Imagick::blurImage -- Adds blur filter to image

Description

bool **Imagick::blurImage** (float \$radius, float \$sigma [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds blur filter to image. Optional third parameter to blur a specific channel.

Parameters

radius

Blur radius

sigma

Standard deviation

channel

The [Channeltype](#) constant. When not supplied, all channels are blurred.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Examples

| Example #9 - Using Imagick::blurImage() : |
|--|
| Blur an image, then display to the browser. <pre><?php header('Content-type: image/jpeg');</pre> |

```
$image = new Imagick('test.jpg');  
  
$image->blurImage(5,3);  
echo $image;  
  
?>
```

See Also

- [Imagick::adaptiveBlurImage\(\)](#)
- [Imagick::motionBlurImage\(\)](#)
- [Imagick::radialBlurImage\(\)](#)

Imagick::borderImage

Imagick::borderImage -- Surrounds the image with a border

Description

bool **Imagick::borderImage** (*mixed* \$bordercolor, int \$width, int \$height)

Warning

This function is currently not documented; only its argument list is available.

Surrounds the image with a border of the color defined by the bordercolor ImagickPixel object.

Parameters

bordercolor

ImagickPixel object or a string containing the border color

width

Border width

height

Border height

Return Values

Returns **TRUE** on success.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as the first parameter. Previous versions allow only an ImagickPixel object. |

Imagick::charcoalImage

Imagick::charcoalImage -- Simulates a charcoal drawing

Description

bool **Imagick::charcoalImage** (float *\$radius*, float *\$sigma*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Simulates a charcoal drawing.

Parameters

radius

The radius of the Gaussian, in pixels, not counting the center pixel

sigma

The standard deviation of the Gaussian, in pixels

Return Values

Returns **TRUE** on success.

Imagick::chopImage

Imagick::chopImage -- Removes a region of an image and trims

Description

bool **Imagick::chopImage** (int \$width, int \$height, int \$x, int \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Removes a region of an image and collapses the image to occupy the removed portion.

Parameters

width

Width of the chopped area

height

Height of the chopped area

x

X origo of the chopped area

y

Y origo of the chopped area

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Examples

| Example #10 - Using Imagick::chopImage() : |
|--|
| Example of using Imagick::chopImage |

```
<?php
/* Create some objects */
$image = new Imagick();
$pixel = new ImagickPixel( 'gray' );

/* New image */
$image->newImage(400, 200, $pixel);

/* Chop image */
$image->chopImage(200, 200, 0, 0);

/* Give image a format */
$image->setImageFormat('png');

/* Output the image with headers */
header('Content-type: image/png');
echo $image;

?>
```

See Also

- [Imagick::cropImage\(\)](#)

Imagick::clear

Imagick::clear -- Clears all resources associated to Imagick object

Description

bool **Imagick::clear** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Clears all resources associated to Imagick object

Return Values

Returns **TRUE** on success.

Imagick::clipImage

Imagick::clipImage -- Clips along the first path from the 8BIM profile

Description

bool **Imagick::clipImage** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Clips along the first path from the 8BIM profile, if present.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::clipPathImage

Imagick::clipPathImage -- Clips along the named paths from the 8BIM profile

Description

bool **Imagick::clipPathImage** (string \$pathname, bool \$inside)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Clips along the named paths from the 8BIM profile, if present. Later operations take effect inside the path. It may be a number if preceded with #, to work on a numbered path, e.g., "#1" to use the first path.

Parameters

pathname

The name of the path

inside

If **TRUE** later operations take effect inside clipping path. Otherwise later operations take effect outside clipping path.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::clone

Imagick::clone -- Makes an exact copy of the Imagick object

Description

[Imagick](#) Imagick::clone (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Makes an exact copy of the Imagick object.

Return Values

What the function returns, first on success, then on failure. See also the &return.success; entity

Imagick::clutImage

Imagick::clutImage -- Replaces colors in the image from a color lookup table

Description

bool **Imagick::clutImage** ([Imagick](#) \$lookup_table [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Replaces colors in the image from a color lookup table. Optional second parameter to replace colors in a specific channel. This method is available if Imagick is compiled against ImageMagick 6.3.5-7 or newer.

Parameters

lookup_table

Imagick object containing the color lookup table

channel

The [Channeltype](#) constant. When not supplied, all channels are replaced.

Return Values

Returns **TRUE** on success.

Examples

| Example #11 - Using Imagick::clutImage() : |
|---|
| Replace colors in the image from a color lookup table. <pre><?php \$image = new Imagick('test.jpg'); \$clut = new Imagick(); \$clut->newImage(1, 1, new ImagickPixel('black')); \$image->clutImage(\$clut); \$image->writeImage('test_out.jpg'); ?></pre> |

See Also

- [Imagick::adaptiveBlurImage\(\)](#)
- [Imagick::motionBlurImage\(\)](#)
- [Imagick::radialBlurImage\(\)](#)

Imagick::coalesceImages

Imagick::coalesceImages -- Composites a set of images

Description

[Imagick](#) Imagick::coalesceImages (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Composites a set of images while respecting any page offsets and disposal methods. GIF, MIFF, and MNG animation sequences typically start with an image background and each subsequent image varies in size and offset. Returns a new Imagick object where each image in the sequence is the same size as the first and composited with the next image in the sequence.

Return Values

Returns a new Imagick object on success, throws ImagickException on failure.

Errors/Exceptions

Throws ImagickException on error.

Imagick::colorFloodfillImage

Imagick::colorFloodfillImage -- Changes the color value of any pixel that matches target

Description

bool **Imagick::colorFloodfillImage** (*mixed* \$fill, float \$fuzz, *mixed* \$bordercolor, int \$x, int \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Changes the color value of any pixel that matches target and is an immediate neighbor.

Parameters

fill

ImagickPixel object containing the fill color

fuzz

The amount of fuzz. For example, set fuzz to 10 and the color red at intensities of 100 and 102 respectively are now interpreted as the same color for the purposes of the floodfill.

bordercolor

ImagickPixel object containing the border color

x

X start position of the floodfill

y

Y start position of the floodfill

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| | |
|--|--|
| | |
|--|--|

| Version | Description |
|---------|--|
| 2.1.0 | Now allows a string representing the color as first and third parameter. Previous versions allow only an MagickPixel object. |

Imagick::colorizeImage

Imagick::colorizeImage -- Blends the fill color with the image

Description

bool **Imagick::colorizeImage** (*mixed* \$colorize, *mixed* \$opacity)

Warning

This function is currently not documented; only its argument list is available.

Blends the fill color with each pixel in the image.

Parameters

colorize

ImagickPixel object or a string containing the colorize color

opacity

ImagickPixel object or an float containing the opacity value. 1.0 is fully opaque and 0.0 is fully transparent.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as the first parameter and a float representing the opacity value as the second parameter. Previous versions allow only an ImagickPixel objects. |

Imagick::combineImages

Imagick::combineImages -- Combines one or more images into a single image

Description

[Imagick](#) **Imagick::combineImages** (int \$channelType)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Combines one or more images into a single image. The grayscale value of the pixels of each image in the sequence is assigned in order to the specified channels of the combined image. The typical ordering would be image 1 => Red, 2 => Green, 3 => Blue, etc.

Parameters

channelType

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::commentImage

Imagick::commentImage -- Adds a comment to your image

Description

bool **Imagick::commentImage** (string *\$comment*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds a comment to your image.

Parameters

comment

The comment to add

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Examples

| Example #12 - Using Imagick::commentImage() : |
|---|
| Commenting an image and retrieving the comment: <pre><?php /* Create new Imagick object */ \$im = new imagick(); /* Create an empty image */ \$im->newImage(100, 100, new ImagickPixel("red")); /* Add comment to the image */ \$im->commentImage("Hello World!");</pre> |


```
/* Display the comment */  
echo $im->getImageProperty("comment");  
  
?>
```

Imagick::compareImageChannels

Imagick::compareImageChannels -- Returns the difference in one or more images

Description

[Imagick](#) Imagick::compareImageChannels ([Imagick](#) \$image, int \$channelType, int \$metricType)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Compares one or more images and returns the difference image.

Parameters

image

Imagick object containing the image to compare.

channelType

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

metricType

One of the [metric type constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::compareImageLayers

Imagick::compareImageLayers -- Returns the maximum bounding region between images

Description

[Imagick](#) Imagick::compareImageLayers (int *\$method*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Compares each image with the next in a sequence and returns the maximum bounding region of any pixel differences it discovers.

Parameters

method

One of the [layer method constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::compareImages

Imagick::compareImages -- Compares an image to a reconstructed image

Description

array **Imagick::compareImages** ([Imagick](#) \$compare, int \$metric)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns an array containing a reconstructed image and the difference between images.

Parameters

compare

An image to compare to.

metric

Provide a valid metric type constant. Refer to this list of [metric constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Examples

| Example #13 - Using Imagick::compareImages() : |
|---|
| Compare images and display the reconstructed image <pre><?php \$image1 = new imagick("image1.png"); \$image2 = new imagick("image2.png"); \$result = \$image1->compareImage(\$image2, Imagick::METRIC_MEANSQUAREERROR); \$result[0]->setImageFormat("png");</pre> |

```
header("Content-Type: image/png");  
echo $result[0];  
  
?>
```

Imagick::compositeImage

Imagick::compositeImage -- Composite one image onto another

Description

bool **Imagick::compositeImage** ([Imagick](#) \$composite_object, int \$composite, int \$x, int \$y [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Composite one image onto another at the specified offset.

Parameters

composite_object

Imagick object which holds the composite image

compose

Composite operator. See [Composite Operator Constants](#)

x

The column offset of the composited image

y

The row offset of the composited image

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Imagick::__construct

Imagick::__construct -- The Imagick constructor

Description

[Imagick](#) Imagick::__construct ([[mixed](#) \$files])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

The Imagick constructor

Parameters

files

The path to an image to load or array of paths

Return Values

Returns a new Imagick object on success, throws ImagickException on failure.

Imagick::contrastImage

Imagick::contrastImage -- Change the contrast of the image

Description

bool **Imagick::contrastImage** (bool *\$sharpen*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Enhances the intensity differences between the lighter and darker elements of the image. Set sharpen to a value other than 0 to increase the image contrast otherwise the contrast is reduced.

Parameters

sharpen

The sharpen value

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::contrastStretchImage

Imagick::contrastStretchImage -- Enhances the contrast of a color image

Description

bool **Imagick::contrastStretchImage** (float \$black_point, float \$white_point [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Enhances the contrast of a color image by adjusting the pixels color to span the entire range of colors available

Parameters

black_point
The black point.

white_point
The white point.

channel
Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Imagick::convolveImage

Imagick::convolveImage -- Applies a custom convolution kernel to the image

Description

bool **Imagick::convolveImage** (array \$kernel [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Applies a custom convolution kernel to the image.

Parameters

kernel

The convolution kernel

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::cropImage

Imagick::cropImage -- Extracts a region of the image

Description

bool **Imagick::cropImage** (int \$width, int \$height, int \$x, int \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Extracts a region of the image.

Parameters

width

The width of the crop

height

The height of the crop

x

The X coordinate of the cropped region's top left corner

y

The Y coordinate of the cropped region's top left corner

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::cropThumbnaillmage

Imagick::cropThumbnaillmage -- Creates a crop thumbnail

Description

bool **Imagick::cropThumbnaillmage** (int \$width, int \$height)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Creates a fixed size thumbnail by first scaling the image down and cropping a specified area from the center.

Parameters

width

The width of the thumbnail

height

The Height of the thumbnail

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::current

Imagick::current -- Returns a reference to the current Imagick object

Description

[Imagick](#) Imagick::current (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns reference to the current imagick object with image pointer at the correct sequence.

Return Values

Returns self on success, throws ImagickException on failure.

Imagick::cycleColormapImage

Imagick::cycleColormapImage -- Displaces an image's colormap

Description

bool **Imagick::cycleColormapImage** (int *\$displace*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Displaces an image's colormap by a given number of positions. If you cycle the colormap a number of times you can produce a psychedelic effect.

Parameters

displace

The amount to displace the colormap.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::deconstructImages

Imagick::deconstructImages -- Returns certain pixel differences between images

Description

bool **Imagick::deconstructImages** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Compares each image with the next in a sequence and returns the maximum bounding region of any pixel differences it discovers.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::despeckleImage

Imagick::despeckleImage -- Reduces the speckle noise in an image

Description

bool **Imagick::despeckleImage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Reduces the speckle noise in an image while perserving the edges of the original image.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::destroy

Imagick::destroy -- Destroys the Imagick object

Description

bool **Imagick::destroy** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Destroys the Imagick object and frees all resources associated with it.

Return Values

Returns **TRUE** on success.

Imagick::displayImage

Imagick::displayImage -- Displays an image

Description

bool **Imagick::displayImage** (string \$servername)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

This method displays an image on a X server.

Parameters

servername

The X server name

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::displayImages

Imagick::displayImages -- Displays an image or image sequence

Description

bool **Imagick::displayImages** (string \$servername)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Displays an image or image sequence on a X server.

Parameters

servername

The X server name

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::distortImage

Imagick::distortImage -- Distorts an image using various distortion methods

Description

bool Imagick::distortImage (int \$method, array \$arguments, bool \$bestfit)

Distorts an image using various distortion methods, by mapping color lookups of the source image to a new destination image usually of the same size as the source image, unless 'bestfit' is set to **TRUE**.

If 'bestfit' is enabled, and distortion allows it, the destination image is adjusted to ensure the whole source 'image' will just fit within the final destination image, which will be sized and offset accordingly. Also in many cases the virtual offset of the source image will be taken into account in the mapping.

This functionality is present if Imagick is compiled against ImageMagick 6.3.6 or later.

Parameters

method

The method of image distortion. See [distortion constants](#)

arguments

The arguments for this distortion method

bestfit

Attempt to resize destination to fit distorted source

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Examples

| |
|---|
| Example #14 - Using Imagick::distortImage(): |
| Distort an image and write it to the disk. |

```
<?php

$im = new imagick( "example.jpg" );

$im->distortImage( Imagick::DISTORTION_PERSPECTIVE, array( 7,40, 4,30,
4,124, 4,123, 85,122, 100,123, 85,2, 100,30 ), true );

$im->writeImage( "example_out.jpg" );

?>
```

See Also

- [Imagick::blurImage\(\)](#)
- [Imagick::motionBlurImage\(\)](#)
- [Imagick::radialBlurImage\(\)](#)

Imagick::drawImage

Imagick::drawImage -- Renders the ImagickDraw object on the current image

Description

bool **Imagick::drawImage** ([ImagickDraw](#) \$draw)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Renders the ImagickDraw object on the current image.

Parameters

draw

The drawing operations to render on the image.

Return Values

Returns **TRUE** on success.

Imagick::edgelImage

Imagick::edgelImage -- Enhance edges within the image

Description

bool **Imagick::edgelImage** (float \$radius)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Enhance edges within the image with a convolution filter of the given radius. Use radius 0 and it will be auto-selected.

Parameters

radius

The radius of the operation.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::embossImage

Imagick::embossImage -- Returns a grayscale image with a three-dimensional effect

Description

bool **Imagick::embossImage** (float \$radius, float \$sigma)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns a grayscale image with a three-dimensional effect. We convolve the image with a Gaussian operator of the given radius and standard deviation (sigma). For reasonable results, radius should be larger than sigma. Use a radius of 0 and it will choose a suitable radius for you.

Parameters

radius

The radius of the effect

sigma

The sigma of the effect

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::enhanceImage

Imagick::enhanceImage -- Improves the quality of a noisy image

Description

bool **Imagick::enhanceImage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Applies a digital filter that improves the quality of a noisy image.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::equalizeImage

Imagick::equalizeImage -- Equalizes the image histogram

Description

bool **Imagick::equalizeImage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Equalizes the image histogram.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::evaluateImage

Imagick::evaluateImage -- Applies an expression to an image

Description

bool **Imagick::evaluateImage** (int *\$op*, float *\$constant* [, int *\$channel*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Applies an arithmetic, relational, or logical expression to an image. Use these operators to lighten or darken an image, to increase or decrease contrast in an image, or to produce the "negative" of an image.

Parameters

op

The operator

constant

The value of the operator

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::flattenImages

Imagick::flattenImages -- Merges a sequence of images

Description

[Imagick](#) Imagick::flattenImages (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Merges a sequence of images. This is useful for combining Photoshop layers into a single image.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::flipImage

Imagick::flipImage -- Creates a vertical mirror image

Description

bool **Imagick::flipImage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Creates a vertical mirror image by reflecting the pixels around the central x-axis.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::flopImage

Imagick::flopImage -- Creates a horizontal mirror image

Description

bool **Imagick::flopImage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Creates a horizontal mirror image by reflecting the pixels around the central y-axis.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::frameImage

Imagick::frameImage -- Adds a simulated three-dimensional border

Description

bool **Imagick::frameImage** ([mixed](#) \$matte_color, int \$width, int \$height, int \$inner_bevel, int \$outer_bevel)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds a simulated three-dimensional border around the image. The width and height specify the border width of the vertical and horizontal sides of the frame. The inner and outer bevels indicate the width of the inner and outer shadows of the frame.

Parameters

matte_color

ImagickPixel object or a string representing the matte color

width

The width of the border

height

The height of the border

inner_bevel

The inner bevel width

outer_bevel

The outer bevel width

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| | |
|--|--|
| | |
|--|--|

| Version | Description |
|---------|--|
| 2.1.0 | Now allows a string representing the color as the first parameter. Previous versions allow only an MagickPixel object. |

Imagick::fxImage

Imagick::fxImage -- Evaluate expression for each pixel in the image

Description

[Imagick](#) **Imagick::fxImage** (string *\$expression* [, int *\$channel*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Evaluate expression for each pixel in the image. Consult [» The Fx Special Effects Image Operator](#) for more information.

Parameters

expression
The expression.

channel
Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::gammaImage

Imagick::gammaImage -- Gamma-corrects an image

Description

bool **Imagick::gammaImage** (float *\$gamma* [, int *\$channel*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Gamma-corrects an image. The same image viewed on different devices will have perceptual differences in the way the image's intensities are represented on the screen. Specify individual gamma levels for the red, green, and blue channels, or adjust all three with the gamma parameter. Values typically range from 0.8 to 2.3.

Parameters

gamma

The amount of gamma-correction.

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::gaussianBlurImage

Imagick::gaussianBlurImage -- Blurs an image

Description

bool **Imagick::gaussianBlurImage** (float \$radius, float \$sigma [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Blurs an image. We convolve the image with a Gaussian operator of the given radius and standard deviation (sigma). For reasonable results, the radius should be larger than sigma. Use a radius of 0 and selects a suitable radius for you.

Parameters

radius

The radius of the Gaussian, in pixels, not counting the center pixel.

sigma

The standard deviation of the Gaussian, in pixels.

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getCompression

Imagick::getCompression -- Gets the object compression type

Description

int **Imagick::getCompression** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the object compression type.

Return Values

Returns **TRUE** on success.

Imagick::getCompressionQuality

Imagick::getCompressionQuality -- Gets the object compression quality

Description

int **Imagick::getCompressionQuality** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the object compression quality.

Return Values

Returns **TRUE** on success.

Imagick::getCopyright

Imagick::getCopyright -- Returns the ImageMagick API copyright as a string

Description

string **Imagick::getCopyright** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the ImageMagick API copyright as a string.

Return Values

Returns a string containing the copyright notice of Imagemagick and Magickwand C API.

Imagick::getFilename

Imagick::getFilename -- The filename associated with an image sequence

Description

string **Imagick::getFilename** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the filename associated with an image sequence.

Return Values

Returns a string on success, throws ImagickException on failure.

Imagick::getFormat

Imagick::getFormat -- Returns the format of the Imagick object

Description

string **Imagick::getFormat** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the format of the Imagick object.

Return Values

Returns the format of the image and throws an ImagickException on failure.

Imagick::getHomeURL

Imagick::getHomeURL -- Returns the ImageMagick home URL

Description

string **Imagick::getHomeURL** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the ImageMagick home URL.

Return Values

Returns a link to the imagemagick homepage.

Imagick::getImage

Imagick::getImage -- Returns a new Imagick object

Description

[Imagick](#) Imagick::getImage (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns a new Imagick object with the current image sequence.

Return Values

Returns a new Imagick object with the current image sequence, throwing ImagickException on error.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageBackgroundColor

Imagick::getImageBackgroundColor -- Returns the image background color

Description

[ImagickPixel](#) Imagick::getImageBackgroundColor (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the image background color.

Return Values

Returns an ImagickPixel set to the background color of the image, throws ImagickException on failure.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageBlob

Imagick::getImageBlob -- Returns the image sequence as a blob

Description

string **Imagick::getImageBlob** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Implements direct to memory image formats. It returns the image sequence as a string. The format of the image determines the format of the returned blob (GIF, JPEG, PNG, etc.). To return a different image format, use Imagick::setImageFormat().

Return Values

Returns a string containing the image. On failure, throws ImagickException.

Imagick::getImageBluePrimary

Imagick::getImageBluePrimary -- Returns the chromaticity blue primary point

Description

[ImagickPixel](#) Imagick::getImageBluePrimary (float \$x, float \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the chromaticity blue primary point for the image.

Parameters

x
The chromaticity blue primary x-point.

y
The chromaticity blue primary x-point.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageBorderColor

Imagick::getImageBorderColor -- Returns the image border color

Description

[ImagickPixel](#) Imagick::getImageBorderColor (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the image border color.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageChannelDepth

Imagick::getImageChannelDepth -- Gets the depth for a particular image channel

Description

int **Imagick::getImageChannelDepth** (int \$channelType)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Gets the depth for a particular image channel.

Parameters

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Imagick::getImageChannelDistortion

Imagick::getImageChannelDistortion -- Compares image channels of an image to a reconstructed image

Description

float **Imagick::getImageChannelDistortion** ([Imagick](#) \$reference, int \$channel, int \$metric)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Compares one or more image channels of an image to a reconstructed image and returns the specified distortion metric.

Parameters

reference

Imagick object to compare to.

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

metric

One of the [metric type constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageChannelExtrema

Imagick::getImageChannelExtrema -- Gets the extrema for one or more image channels

Description

array **Imagick::getImageChannelExtrema** (int `$channel`)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Gets the extrema for one or more image channels. Return value is an associative array with the keys "minima" and "maxima".

Parameters

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageChannelMean

Imagick::getImageChannelMean -- Gets the mean and standard deviation

Description

array **Imagick::getImageChannelMean** (int *\$channel*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Gets the mean and standard deviation of one or more image channels. Return value is an associative array with the keys "mean" and "standardDeviation".

Parameters

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageChannelStatistics

Imagick::getImageChannelStatistics -- Returns statistics for each channel in the image

Description

array **Imagick::getImageChannelStatistics** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns statistics for each channel in the image. The statistics include the channel depth, its minima and maxima, the mean, and the standard deviation. You can access the red channel mean, for example, like this:

Return Values

Returns **TRUE** on success.

Imagick::getImageColormapColor

Imagick::getImageColormapColor -- Returns the color of the specified colormap index

Description

[ImagickPixel](#) Imagick::getImageColormapColor (int *\$index*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the color of the specified colormap index.

Parameters

index

The offset into the image colormap.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageColors

Imagick::getImageColors -- Gets the number of unique colors in the image

Description

int **Imagick::getImageColors** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the number of unique colors in the image.

Return Values

Returns **TRUE** on success.

Imagick::getImageColorspace

Imagick::getImageColorspace -- Gets the image colorspace

Description

int **Imagick::getImageColorspace** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image colorspace.

Return Values

Returns **TRUE** on success.

Imagick::getImageCompose

Imagick::getImageCompose -- Returns the composite operator associated with the image

Description

int **Imagick::getImageCompose** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the composite operator associated with the image.

Return Values

Returns **TRUE** on success.

Imagick::getImageDelay

Imagick::getImageDelay -- Gets the image delay

Description

int **Imagick::getImageDelay** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image delay.

Return Values

Returns the image delay and throws ImagickException on failure.

Imagick::getImageDepth

Imagick::getImageDepth -- Gets the image depth

Description

int **Imagick::getImageDepth** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image depth.

Return Values

What the function returns, first on success, then on failure. See also the &return.success; entity

Imagick::getImageDispose

Imagick::getImageDispose -- Gets the image disposal method

Description

int **Imagick::getImageDispose** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image disposal method.

Return Values

Returns the dispose method on success, throws an ImagickException on failure.

Imagick::getImageDistortion

Imagick::getImageDistortion -- Compares an image to a reconstructed image

Description

float **Imagick::getImageDistortion** ([MagickWand](#) \$reference, int \$metric)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Compares an image to a reconstructed image and returns the specified distortion metric.

Parameters

reference

Imagick object to compare to.

metric

One of the [metric type constants](#).

Return Values

Returns the distortion metric used on the image (or the best guess thereof). Throws an exception on error.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageExtrema

Imagick::getImageExtrema -- Gets the extrema for the image

Description

array **Imagick::getImageExtrema** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Gets the extrema for the image. Returns an associative array with the keys "min" and "max".

Return Values

Returns an associative array with the keys "min" and "max". Throws ImagickException on failure.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageFilename

Imagick::getImageFilename -- Returns the filename of a particular image in a sequence

Description

string **Imagick::getImageFilename** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the filename of a particular image in a sequence.

Return Values

Returns a string with the filename of the image. Throws an ImagickException on failure.

Imagick::getImageFormat

Imagick::getImageFormat -- Returns the format of a particular image in a sequence

Description

string **Imagick::getImageFormat** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the format of a particular image in a sequence.

Return Values

Returns a string containing the image format on success, throws ImagickException on failure.

Imagick::getImageGamma

Imagick::getImageGamma -- Gets the image gamma

Description

float **Imagick::getImageGamma** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image gamma.

Return Values

Returns the image gamma on success, throws ImagickException on failure.

Imagick::getImageGeometry

Imagick::getImageGeometry -- Gets the width and height as an associative array

Description

array **Imagick::getImageGeometry** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the width and height as an associative array.

Return Values

Returns an array with the width/height of the image, throws ImagickException on error.

Imagick::getImageGreenPrimary

Imagick::getImageGreenPrimary -- Returns the chromaticity green primary point

Description

array **Imagick::getImageGreenPrimary** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the chromaticity green primary point. Returns an array with the keys "x" and "y".

Return Values

Returns an array with the keys "x" and "y" on success, throws an ImagickException on failure.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageHeight

Imagick::getImageHeight -- Returns the image height

Description

int **Imagick::getImageHeight** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the image height.

Return Values

Returns the image height in pixels, throws ImagickException on error.

Imagick::getImageHistogram

Imagick::getImageHistogram -- Gets the image histogram

Description

array **Imagick::getImageHistogram** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the image histogram as an array of ImagickPixel objects.

Return Values

Returns the image histogram as an array of ImagickPixel objects, throwing ImagickException on error.

Imagick::getImageIndex

Imagick::getImageIndex -- Gets the index of the current active image

Description

int **Imagick::getImageIndex** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the index of the current active image within the Imagick object. This method has been deprecated. See [Imagick::getImageIteratorIndex](#)

Return Values

Returns an integer containing the index of the image in the stack, throwing ImagickException on error.

Imagick::getImageInterlaceScheme

Imagick::getImageInterlaceScheme -- Gets the image interlace scheme

Description

int **Imagick::getImageInterlaceScheme** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image interlace scheme.

Return Values

Returns the interlace scheme as an integer on success, throwing ImagickException on failure.

Imagick::getImageInterpolateMethod

Imagick::getImageInterpolateMethod -- Returns the interpolation method

Description

int **Imagick::getImageInterpolateMethod** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the interpolation method for the sepcified image.

Return Values

Returns the interpolate method on success, throws ImagickException on error.

Imagick::getImageIterations

Imagick::getImageIterations -- Gets the image iterations

Description

int **Imagick::getImageIterations** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image iterations.

Return Values

Returns the image iterations as an integer, throws ImagickException on failure.

Imagick::getImageLength

Imagick::getImageLength -- Returns the image length in bytes

Description

int **Imagick::getImageLength** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the image length in bytes

Return Values

Returns an int containing the current image size.

Imagick::getImageMagickLicense

Imagick::getImageMagickLicense -- Returns a string containing the ImageMagick license

Description

string **Imagick::getImageMagickLicense** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns a string containing the ImageMagick license

Return Values

Returns a string containing the ImageMagick license.

Imagick::getImageMatte

Imagick::getImageMatte -- Return if the image has a matte channel

Description

int **Imagick::getImageMatte** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns **TRUE** if the image has a matte channel otherwise false.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Imagick::getImageMatteColor

Imagick::getImageMatteColor -- Returns the image matte color

Description

[ImagickPixel](#) Imagick::getImageMatteColor (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the image matte color.

Return Values

Returns ImagickPixel object on success and throws ImagickException on failure.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageOrientation

Imagick::getImageOrientation -- Gets the image orientation.

Description

int **Imagick::getImageOrientation** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image orientation. The return value is one of the [orientation constants](#).

Return Values

Returns an int on success, throws ImagickException on failure.

Imagick::getImagePage

Imagick::getImagePage -- Returns the page geometry

Description

array **Imagick::getImagePage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the page geometry associated with the image in an array with the keys "width", "height", "x", and "y".

Return Values

Returns the page geometry associated with the image in an array with the keys "width", "height", "x", and "y".

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImagePixelColor

Imagick::getImagePixelColor -- Returns the color of the specified pixel

Description

[ImagickPixel](#) Imagick::getImagePixelColor (int x , int y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the color of the specified pixel.

Parameters

x

The x-coordinate of the pixel

y

The y-coordinate of the pixel

Return Values

Returns an ImagickPixel instance for the color at the coordinates given, throws ImagickException on error.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageProfile

Imagick::getImageProfile -- Returns the named image profile

Description

string **Imagick::getImageProfile** (string *\$name*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the named image profile.

Parameters

name

The name of the profile to return.

Return Values

Returns a string containing the image profile, throws ImagickException on error.

Imagick::getImageProfiles

Imagick::getImageProfiles -- Returns the image profiles

Description

array **Imagick::getImageProfiles** ([string \$pattern [, bool \$only_names]])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns all associated profiles that match the pattern. If **TRUE** is passed as second parameter only the profile names are returned. This method is present if Imagick is compiled against ImageMagick 6.3.5-9 or later.

Parameters

pattern

The pattern for profile names. Defaults to ""

only_names

Whether to return only profile names

Return Values

Returns an array containing the image profiles or profile names.

Imagick::getImageProperties

Imagick::getImageProperties -- Returns the image properties

Description

array **Imagick::getImageProperties** ([string *\$pattern* [, bool *\$only_names*]])

Warning

This function is currently not documented; only its argument list is available.

Returns all associated properties that match the pattern. If **TRUE** is passed as second parameter only the property names are returned. This method is present if Imagick is compiled against ImageMagick 6.3.5-9 or later.

Parameters

pattern

The pattern for property names. Defaults to ""

only_names

Whether to return only property names

Return Values

Returns an array containing the image properties or property names.

Examples

Example #15 - Using [Imagick::getImageProperties\(\)](#):

An example of extracting EXIF information.

```
<?php

/* Create the object */
$im = new imagick("/path/to/example.jpg");

/* Get the EXIF information */
$exifArray = $im->getImageProperties("exif:*");

/* Loop through the exif properties */
foreach ($exifArray as $name => $property)
```

```
{
  echo "{$name} => {$property}<br />\n";
}

?>
```

Imagick::getImageProperty

Imagick::getImageProperty -- Returns the named image property

Description

string **Imagick::getImageProperty** (string *\$name*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the named image profile.

Parameters

name

name of the property (for example Exif:DateTime)

Return Values

Returns a string containing the image property, false if a property with the given name does not exist.

Imagick::getImageRedPrimary

Imagick::getImageRedPrimary -- Returns the chromaticy red primary point

Description

array **Imagick::getImageRedPrimary** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the chromaticy red primary point as an array with the keys "x" and "y".

Return Values

Returns the chromaticy red primary point as an array with the keys "x" and "y", throwing ImagickException on error.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageRegion

Imagick::getImageRegion -- Extracts a region of the image

Description

[Imagick](#) **Imagick::getImageRegion** (int \$width, int \$height, int \$x, int \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Extracts a region of the image and returns it as a new Imagick object.

Parameters

width

The width of the extracted region.

height

The height of the extracted region.

x

X-coordinate of the top-left corner of the extracted region.

y

Y-coordinate of the top-left corner of the extracted region.

Return Values

Extracts a region of the image and returns it as a new wand, throwing ImagickException on error.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageRenderingIntent

Imagick::getImageRenderingIntent -- Gets the image rendering intent

Description

int **Imagick::getImageRenderingIntent** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image rendering intent.

Return Values

Returns the image [rendering intent](#), throwing ImagickException on error.

Imagick::getImageResolution

Imagick::getImageResolution -- Gets the image X and Y resolution

Description

array **Imagick::getImageResolution** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image X and Y resolution.

Return Values

Returns the resolution as an array and throws ImagickException on error.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageScene

Imagick::getImageScene -- Gets the image scene

Description

int **Imagick::getImageScene** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image scene.

Return Values

Returns the image scene, throwing ImagickException on error.

Imagick::getImageSignature

Imagick::getImageSignature -- Generates an SHA-256 message digest

Description

string **Imagick::getImageSignature** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Generates an SHA-256 message digest for the image pixel stream.

Return Values

Returns a string containing the SHA-256 hash of the file, throwing an ImagickException on error.

Imagick::getImageSize

Imagick::getImageSize -- Returns the image length in bytes

Description

int **Imagick::getImageSize** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the image length in bytes

Return Values

Returns an int containing the current image size.

Imagick::getImageTicksPerSecond

Imagick::getImageTicksPerSecond -- Gets the image ticks-per-second

Description

int **Imagick::getImageTicksPerSecond** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image ticks-per-second.

Return Values

Returns the image ticks-per-second, throwing ImagickException on error.

Imagick::getImageTotalInkDensity

Imagick::getImageTotalInkDensity -- Gets the image total ink density

Description

float **Imagick::getImageTotalInkDensity** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image total ink density.

Return Values

Returns the image total ink density of the image, throwing ImagickException on error.

Imagick::getImageType

Imagick::getImageType -- Gets the potential image type

Description

int **Imagick::getImageType** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the potential image type.

Return Values

Retruns the potential image type, throwing ImagickException on error.

Imagick::getImageUnits

Imagick::getImageUnits -- Gets the image units of resolution

Description

int **Imagick::getImageUnits** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the image units of resolution.

Return Values

Returns the image units of resolution, throwing ImagickException on failure.

Imagick::getImageVirtualPixelMethod

Imagick::getImageVirtualPixelMethod -- Returns the virtual pixel method

Description

int **Imagick::getImageVirtualPixelMethod** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the virtual pixel method for the specified image.

Return Values

Returns the virtual pixel method on success, ImagickException on failure.

Imagick::getImageWhitePoint

Imagick::getImageWhitePoint -- Returns the chromaticy white point

Description

array **Imagick::getImageWhitePoint** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the chromaticy white point as an associative array with the keys "x" and "y".

Return Values

Returns the chromaticy white point as an associative array with the keys "x" and "y", throws ImagickException on error.

Errors/Exceptions

Throws ImagickException on error.

Imagick::getImageWidth

Imagick::getImageWidth -- Returns the image width

Description

int **Imagick::getImageWidth** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the image width.

Return Values

Returns the image width, throwing ImagickException on error.

Imagick::getInterlaceScheme

Imagick::getInterlaceScheme -- Gets the object interlace scheme

Description

int **Imagick::getInterlaceScheme** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the object interlace scheme.

Return Values

Gets the wand [interlace scheme](#), throwing ImagickException on error.

Imagick::getIteratorIndex

Imagick::getIteratorIndex -- Gets the index of the current active image

Description

int **Imagick::getIteratorIndex** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the index of the current active image within the Imagick object.

Return Values

Returns an integer containing the index of the image in the stack, throwing ImagickException on error.

Imagick::getNumberImages

Imagick::getNumberImages -- Returns the number of images in the object

Description

int **Imagick::getNumberImages** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the number of images associated with Imagick object.

Return Values

Returns the number of images associated with Imagick object, throwing ImagickException on failure.

Imagick::getOption

Imagick::getOption -- Returns a value associated with the specified key

Description

string **Imagick::getOption** (string *\$key*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns a value associated within the object for the specified key.

Parameters

key

The name of the option

Return Values

Returns a value associated with a wand and the specified key, throwing ImagickException on error.

Imagick::getPackageName

Imagick::getPackageName -- Returns the ImageMagick package name

Description

string **Imagick::getPackageName** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the ImageMagick package name.

Return Values

Returns the ImageMagick package name as a string, throwing ImagickException on error.

Imagick::getPage

Imagick::getPage -- Returns the page geometry

Description

array **Imagick::getPage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the page geometry associated with the Imagick object in an associative array with the keys "width", "height", "x", and "y".

Return Values

Returns the page geometry associated with the Imagick object in an associative array with the keys "width", "height", "x", and "y", throwing ImagickException on error.

Imagick::getPixelIterator

Imagick::getPixelIterator -- Returns a MagickPixelIterator

Description

[ImagickPixelIterator](#) Imagick::getPixelIterator (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns a MagickPixelIterator.

Return Values

Returns an ImagickPixelIterator on success, throwing ImagickException on failure.

Imagick::getPixelRegionIterator

Imagick::getPixelRegionIterator -- Get an ImagickPixelIterator for an image section

Description

[ImagickPixelIterator](#) **Imagick::getPixelRegionIterator** (int *\$x*, int *\$y*, int *\$columns*, int *\$rows*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Get an ImagickPixelIterator for an image section.

Parameters

x
The x-coordinate of the region.

y
The y-coordinate of the region.

columns
The width of the region.

rows
The height of the region.

Return Values

Returns an ImagickPixelIterator for an image section, throwing ImagickException on error.

Imagick::getQuantumDepth

Imagick::getQuantumDepth -- Gets the quantum depth

Description

array **Imagick::getQuantumDepth** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the Imagick quantum depth as a string.

Return Values

Returns the Imagick quantum depth as a string, throwing ImagickException on error.

Imagick::getQuantumRange

Imagick::getQuantumRange -- Returns the Imagick quantum range

Description

array **Imagick::getQuantumRange** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the Imagick quantum range as a string.

Return Values

Returns the Imagick quantum range as a string, throwing ImagickException on error.

Imagick::getReleaseDate

Imagick::getReleaseDate -- Returns the ImageMagick release date

Description

string **Imagick::getReleaseDate** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the ImageMagick release date as a string.

Return Values

Returns the ImageMagick release date as a string, throwing ImagickException on error.

Imagick::getResource

Imagick::getResource -- Returns the specified resource's memory usage

Description

int **Imagick::getResource** (int *\$type*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the specified resource's memory usage in megabytes.

Parameters

type

Refer to the list of [resourcetype constants](#).

Return Values

Returns the specified resource's memory usage in megabytes, throwing ImagickException on error.

Imagick::getResourceLimit

Imagick::getResourceLimit -- Returns the specified resource limit

Description

int **Imagick::getResourceLimit** (int *\$type*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the specified resource limit in megabytes.

Parameters

type

Refer to the list of [resourcetype constants](#).

Return Values

Returns the specified resource limit in megabytes, throwing ImagickException on error.

Imagick::getSamplingFactors

Imagick::getSamplingFactors -- Gets the horizontal and vertical sampling factor

Description

array **Imagick::getSamplingFactors** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Gets the horizontal and vertical sampling factor.

Return Values

Returns an associative array with the horizontal and vertical sampling factors of the image.
Throws ImagickException on error.

Imagick::getSize

Imagick::getSize -- Returns the size associated with the Imagick object

Description

array **Imagick::getSize** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the size associated with the Imagick object as an array with the keys "columns" and "rows".

Return Values

Returns the size associated with the Imagick object as an array with the keys "columns" and "rows".

Imagick::getSizeOffset

Imagick::getSizeOffset -- Returns the size offset

Description

int **Imagick::getSizeOffset** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the size offset associated with the Imagick object.

Return Values

Returns the size offset associated with the Imagick object, throwing ImagickException on error.

Imagick::getVersion

Imagick::getVersion -- Returns the ImageMagick API version

Description

array **Imagick::getVersion** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the ImageMagick API version as a string and as a number.

Return Values

Returns the ImageMagick API version as a string and as a number, throwing ImagickException on error.

Imagick::hasNextImage

Imagick::hasNextImage -- Checks if the object has more images

Description

bool **Imagick::hasNextImage** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns **TRUE** if the object has more images when traversing the list in the forward direction.

Return Values

Returns **TRUE** if the object has more images when traversing the list in the forward direction, returns **FALSE** if there are none.

Imagick::hasPreviousImage

Imagick::hasPreviousImage -- Checks if the object has a previous image

Description

bool **Imagick::hasPreviousImage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns **TRUE** if the object has more images when traversing the list in the reverse direction

Return Values

Returns **TRUE** if the object has more images when traversing the list in the reverse direction, returns **FALSE** if there are none.

Imagick::identifyImage

Imagick::identifyImage -- Identifies an image and fetches attributes

Description

array **Imagick::identifyImage** ([bool \$appendRawOutput])

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Identifies an image and returns the attributes. Attributes include the image width, height, size, and others.

Parameters

appendRawOutput

Return Values

Identifies an image and returns the attributes. Attributes include the image width, height, size, and others. Throws ImagickException on error.

Errors/Exceptions

Throws ImagickException on error.

Imagick::implodeImage

Imagick::implodeImage -- Creates a new image as a copy

Description

bool **Imagick::implodeImage** (float \$radius)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Creates a new image that is a copy of an existing one with the image pixels "imploded" by the specified percentage.

Parameters

radius

The radius of the implode

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::labelImage

Imagick::labelImage -- Adds a label to an image

Description

bool **Imagick::labelImage** (string *\$label*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds a label to an image.

Parameters

label

The label to add

Return Values

Returns **TRUE** on success.

Imagick::levelImage

Imagick::levelImage -- Adjusts the levels of an image

Description

```
bool Imagick::levelImage ( float $blackPoint, float $gamma, float $whitePoint [, int $channel ] )
```

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adjusts the levels of an image by scaling the colors falling between specified white and black points to the full available quantum range. The parameters provided represent the black, mid, and white points. The black point specifies the darkest color in the image. Colors darker than the black point are set to zero. Mid point specifies a gamma correction to apply to the image. White point specifies the lightest color in the image. Colors brighter than the white point are set to the maximum quantum value.

Parameters

blackPoint

The image black point

gamma

The gamma value

whitePoint

The image white point

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::linearStretchImage

Imagick::linearStretchImage -- Stretches with saturation the image intensity

Description

bool **Imagick::linearStretchImage** (float \$blackPoint, float \$whitePoint)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Stretches with saturation the image intensity.

Parameters

blackPoint

The image black point

whitePoint

The image white point

Return Values

Returns **TRUE** on success.

Imagick::magnifyImage

Imagick::magnifyImage -- Scales an image proportionally 2x

Description

bool **Imagick::magnifyImage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Is a convenience method that scales an image proportionally to twice its original size.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::mapImage

Imagick::mapImage -- Replaces the colors of an image with the closest color from a reference image.

Description

bool **Imagick::mapImage** ([Imagick](#) \$map, bool \$dither)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Parameters

map

dither

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::matteFloodfillImage

Imagick::matteFloodfillImage -- Changes the transparency value of a color

Description

bool **Imagick::matteFloodfillImage** (float \$alpha, float \$fuzz, mixed \$bordercolor, int \$x, int \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Changes the transparency value of any pixel that matches target and is an immediate neighbor. If the method FillToBorderMethod is specified, the transparency value is changed for any neighbor pixel that does not match the bordercolor member of image.

Parameters

alpha

The level of transparency: 1.0 is fully opaque and 0.0 is fully transparent.

fuzz

The fuzz member of image defines how much tolerance is acceptable to consider two colors as the same.

bordercolor

An ImagickPixel object or string representing the border color.

x

The starting x coordinate of the operation.

y

The starting y coordinate of the operation.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| | |
|--|--|
| | |
|--|--|

| Version | Description |
|---------|--|
| 2.1.0 | Now allows a string representing the color as the third parameter. Previous versions allow only an MagickPixel object. |

Imagick::medianFilterImage

Imagick::medianFilterImage -- Applies a digital filter

Description

bool **Imagick::medianFilterImage** (float *\$radius*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Applies a digital filter that improves the quality of a noisy image. Each pixel is replaced by the median in a set of neighboring pixels as defined by radius.

Parameters

radius

The radius of the pixel neighborhood.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::minifyImage

Imagick::minifyImage -- Scales an image proportionally to half its size

Description

bool **Imagick::minifyImage** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Is a convenience method that scales an image proportionally to one-half its original size

Return Values

Returns **TRUE** on success.

Imagick::modulateImage

Imagick::modulateImage -- Control the brightness, saturation, and hue

Description

bool **Imagick::modulateImage** (float \$brightness, float \$saturation, float \$hue)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Lets you control the brightness, saturation, and hue of an image. Hue is the percentage of absolute rotation from the current position. For example 50 results in a counter-clockwise rotation of 90 degrees, 150 results in a clockwise rotation of 90 degrees, with 0 and 200 both resulting in a rotation of 180 degrees.

Parameters

brightness

saturation

hue

Return Values

Returns **TRUE** on success.

Imagick::montageImage

Imagick::montageImage -- Creates a composite image

Description

[Imagick](#) **Imagick::montageImage** ([ImagickDraw](#) \$draw, string \$tile_geometry, string \$thumbnail_geometry, int \$mode, string \$frame)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Creates a composite image by combining several separate images. The images are tiled on the composite image with the name of the image optionally appearing just below the individual tile.

Parameters

draw

The font name, size, and color are obtained from this object.

tile_geometry

The number of tiles per row and page (e.g. 6x4+0+0).

thumbnail_geometry

Preferred image size and border size of each thumbnail (e.g. 120x120+4+3>).

mode

Thumbnail framing mode, see [Montage Mode constants](#).

frame

Surround the image with an ornamental border (e.g. 15x15+3+3). The frame color is that of the thumbnail's matte color.

Return Values

Returns **TRUE** on success.

Imagick::morphImages

Imagick::morphImages -- Method morphs a set of images

Description

[Imagick](#) **Imagick::morphImages** (int \$number_frames)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Method morphs a set of images. Both the image pixels and size are linearly interpolated to give the appearance of a meta-morphosis from one image to the next.

Parameters

number_frames

The number of in-between images to generate.

Return Values

This method returns a new Imagick object on success and throws an ImagickException on error.

Imagick::mosaicImages

Imagick::mosaicImages -- Forms a mosaic from images

Description

[Imagick](#) **Imagick::mosaicImages** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Inlays an image sequence to form a single coherent picture. It returns a wand with each image in the sequence composited at the location defined by the page offset of the image.

Return Values

Returns **TRUE** on success.

Imagick::motionBlurImage

Imagick::motionBlurImage -- Simulates motion blur

Description

bool **Imagick::motionBlurImage** (float \$radius, float \$sigma, float \$angle)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Simulates motion blur. We convolve the image with a Gaussian operator of the given radius and standard deviation (sigma). For reasonable results, radius should be larger than sigma. Use a radius of 0 and MotionBlurImage() selects a suitable radius for you. Angle gives the angle of the blurring motion.

Parameters

radius

The radius of the Gaussian, in pixels, not counting the center pixel.

sigma

The standard deviation of the Gaussian, in pixels.

angle

Apply the effect along this angle.

Return Values

Returns **TRUE** on success.

Imagick::negateImage

Imagick::negateImage -- Negates the colors in the reference image

Description

bool **Imagick::negateImage** (bool *\$gray* [, int *\$channel*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Negates the colors in the reference image. The Grayscale option means that only grayscale values within the image are negated.

Parameters

gray

Whether to only negate grayscale pixels within the image.

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::newImage

Imagick::newImage -- Creates a new image

Description

bool **Imagick::newImage** (int \$cols, int \$rows, mixed \$background [, string \$format])

Warning

This function is currently not documented; only its argument list is available.

Creates a new image and associates ImagickPixel value as background color

Parameters

cols

Columns in the new image

rows

Rows in the new image

background

The background color used for this image

format

Image format. This parameter was added in Imagick version 2.0.1.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|--|
| 2.1.0 | Now allows a string representing the color as the third parameter. Previous versions |

| |
|------------------------------------|
| allow only an ImagickPixel object. |
|------------------------------------|

Examples

Example #16 - Using [Imagick::newImage\(\)](#):

Create a new image and display it.

```
<?php

$image = new Imagick();
$image->newImage(100, 100, new ImagickPixel('red'));
$image->setImageFormat('png');

header('Content-type: image/png');
echo $image;

?>
```

Imagick::newPseudoImage

Imagick::newPseudoImage -- Creates a new image

Description

bool **Imagick::newPseudoImage** (int \$columns, int \$rows, string \$pseudoString)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Creates a new image using ImageMagick pseudo-formats.

Parameters

columns

columns in the new image

rows

rows in the new image

pseudoString

string containing pseudo image definition.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::nextImage

Imagick::nextImage -- Moves to the next image

Description

bool **Imagick::nextImage** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Associates the next image in the image list with an Imagick object.

Return Values

Returns **TRUE** on success.

Imagick::normalizeImage

Imagick::normalizeImage -- Enhances the contrast of a color image

Description

bool **Imagick::normalizeImage** ([int *\$channel*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Enhances the contrast of a color image by adjusting the pixels color to span the entire range of colors available.

Parameters

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Imagick::oilPaintImage

Imagick::oilPaintImage -- Simulates an oil painting

Description

bool **Imagick::oilPaintImage** (float *\$radius*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Applies a special effect filter that simulates an oil painting. Each pixel is replaced by the most frequent color occurring in a circular region defined by radius.

Parameters

radius

The radius of the circular neighborhood.

Return Values

Returns **TRUE** on success.

Imagick::optimizeImageLayers

Imagick::optimizeImageLayers -- Removes repeated portions of images to optimize

Description

bool **Imagick::optimizeImageLayers** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Compares each image the GIF disposed forms of the previous image in the sequence. From this it attempts to select the smallest cropped image to replace each frame, while preserving the results of the animation.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::colorFloodfillImage

Imagick::colorFloodfillImage -- Changes the color value of any pixel that matches target

Description

bool **Imagick::paintFloodfillImage** ([mixed](#) \$fill, float \$fuzz, [mixed](#) \$bordercolor, int \$x, int \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Changes the color value of any pixel that matches target and is an immediate neighbor.

Parameters

fill

ImagickPixel object or a string containing the fill color

fuzz

The amount of fuzz. For example, set fuzz to 10 and the color red at intensities of 100 and 102 respectively are now interpreted as the same color for the purposes of the floodfill.

bordercolor

ImagickPixel object or a string containing the border color

x

X start position of the floodfill

y

Y start position of the floodfill

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Imagick::paintOpaqueImage

Imagick::paintOpaqueImage -- Change any pixel that matches color

Description

bool **Imagick::paintOpaqueImage** ([mixed](#) \$target, [mixed](#) \$fill, float \$fuzz [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Changes any pixel that matches color with the color defined by fill.

Parameters

target

Change this target color to the fill color within the image. An ImagickPixel object or a string representing the target color.

fill

An ImagickPixel object or a string representing the fill color.

fuzz

The fuzz member of image defines how much tolerance is acceptable to consider two colors as the same.

channel

Provide any channel constant that is valid for your channel mode. To apply to more than one channel, combine channeltype constants using bitwise operators. Defaults to Imagick::CHANNEL_ALL. Refer to this list of [channel constants](#).

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as first and second parameter. Previous versions allow only an MagickPixel object. |

Imagick::paintTransparentImage

Imagick::paintTransparentImage -- Changes any pixel that matches color with the color defined by fill

Description

bool **Imagick::paintTransparentImage** ([mixed](#) \$target, float \$alpha, float \$fuzz)

Warning

This function is currently not documented; only its argument list is available.

Changes any pixel that matches color with the color defined by fill.

Parameters

target

Change this target color to specified opacity value within the image.

alpha

The level of transparency: 1.0 is fully opaque and 0.0 is fully transparent.

fuzz

The fuzz member of image defines how much tolerance is acceptable to consider two colors as the same.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as the first parameter. Previous versions allow only an ImagickPixel object. |

Imagick::pingImage

Imagick::pingImage -- Fetch basic attributes about the image

Description

bool **Imagick::pingImage** (string *\$filename*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

This method can be used to query image width, height, size, and format without reading the whole image in to memory.

Parameters

filename

The filename to read the information from.

Return Values

Returns **TRUE** on success.

Imagick::pingImageBlob

Imagick::pingImageBlob -- Quickly fetch attributes

Description

bool **Imagick::pingImageBlob** (string *\$image*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

This method can be used to query image width, height, size, and format without reading the whole image to memory.

Parameters

image

A string containing the image.

Return Values

Returns **TRUE** on success.

Imagick::pingImageFile

Imagick::pingImageFile -- Get basic image attributes in a lightweight manner

Description

bool **Imagick::pingImageFile** (resource \$filehandle [, string \$fileName])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

This method can be used to query image width, height, size, and format without reading the whole image to memory.

Parameters

filehandle

An open filehandle to the image.

fileName

Optional filename for this image.

Return Values

Returns **TRUE** on success.

Imagick::polaroidImage

Imagick::polaroidImage -- Simulates a Polaroid picture

Description

bool **Imagick::polaroidImage** ([ImagickDraw](#) \$properties, float \$angle)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Simulates a Polaroid picture. This method is available if you compile Imagick against ImageMagick 6.3.2 or later.

Parameters

properties
The polaroid properties

angle
The polaroid angle

Return Values

Returns **TRUE** on success.

Imagick::posterizeImage

Imagick::posterizeImage -- Reduces the image to a limited number of color level

Description

bool **Imagick::posterizeImage** (int *\$levels*, bool *\$dither*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Reduces the image to a limited number of color level.

Parameters

levels

dither

Return Values

Returns **TRUE** on success.

Imagick::previewImages

Imagick::previewImages -- Quickly pin-point appropriate parameters for image processing

Description

bool **Imagick::previewImages** (int \$preview)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Tiles 9 thumbnails of the specified image with an image processing operation applied at varying strengths. This is helpful to quickly pin-point an appropriate parameter for an image processing operation.

Parameters

preview

Preview type. See [Preview type constants](#)

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::previousImage

Imagick::previousImage -- Move to the previous image in the object

Description

bool **Imagick::previousImage** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Associates the previous image in an image list with the Imagick object.

Return Values

Returns **TRUE** on success.

Imagick::profileImage

Imagick::profileImage -- Adds or removes a profile from an image

Description

bool **Imagick::profileImage** (string \$name, string \$profile)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds or removes a ICC, IPTC, or generic profile from an image. If the profile is NULL, it is removed from the image otherwise added. Use a name of '*' and a profile of NULL to remove all profiles from the image.

Parameters

name

profile

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::quantizeImage

Imagick::quantizeImage -- Analyzes the colors within a reference image

Description

```
bool Imagick::quantizeImage ( int $numberOfColors, int $colorspace, int $treeDepth, bool  
$dither, bool $measureError )
```

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Parameters

numberOfColors

colorspace

treeDepth

dither

measureError

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::quantizeImages

Imagick::quantizeImages -- Analyzes the colors within a sequence of images

Description

```
bool Imagick::quantizeImages ( int $numberOfColors, int $colorSpace, int $treeDepth,  
bool $dither, bool $measureError )
```

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Parameters

numberOfColors

colorSpace

treeDepth

dither

measureError

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::queryFontMetrics

Imagick::queryFontMetrics -- Returns an array representing the font metrics

Description

array **Imagick::queryFontMetrics** ([ImagickDraw](#) \$properties, string \$text [, bool \$multiline])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns a multi-dimensional array representing the font metrics.

Parameters

properties
ImagickDraw object containing font properties

text
The text

multiline
Multiline parameter. If left empty it is autodetected

Return Values

Returns an array containing the formats supported by Imagick, throws ImagickException on error.

Errors/Exceptions

Throws ImagickException on error.

Examples

| Example #17 - Using Imagick::queryFontMetrics() : |
|---|
| Query the metrics for the text and dump the results on the screen. <pre><?php /* Create a new Imagick object */</pre> |

```
$im = new Imagick();

/* Create an ImagickDraw object */
$draw = new ImagickDraw();

/* Set the font */
$draw->setFont('/path/to/font.ttf');

/* Dump the font metrics, autodetect multiline */
var_dump($im->queryFontMetrics($draw, "Hello World!"));
?>
```

Imagick::queryFonts

Imagick::queryFonts -- Returns the configured fonts

Description

array **Imagick::queryFonts** ([string *\$pattern*])

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns formats supported by Imagick.

Parameters

pattern

The query pattern

Return Values

Returns an array containing the configured fonts, throws ImagickException on error.

Imagick::queryFormats

Imagick::queryFormats -- Returns formats supported by Imagick

Description

array **Imagick::queryFormats** ([string *\$pattern*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns formats supported by Imagick.

Parameters

pattern

Return Values

Returns an array containing the formats supported by Imagick, throws ImagickException on error.

Imagick::radialBlurImage

Imagick::radialBlurImage -- Radial blurs an image

Description

bool **Imagick::radialBlurImage** (float *\$angle* [, int *\$channel*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Radial blurs an image.

Parameters

angle

channel

Return Values

Returns **TRUE** on success.

Imagick::raiseImage

Imagick::raiseImage -- Creates a simulated 3d button-like effect

Description

bool **Imagick::raiseImage** (int \$width, int \$height, int \$x, int \$y, bool \$raise)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Creates a simulated three-dimensional button-like effect by lightening and darkening the edges of the image. Members width and height of raise_info define the width of the vertical and horizontal edge of the effect.

Parameters

width

height

x

y

raise

Return Values

Returns **TRUE** on success.

Imagick::randomThresholdImage

Imagick::randomThresholdImage -- Creates a high-contrast, two-color image

Description

bool **Imagick::randomThresholdImage** (float *\$low*, float *\$high* [, int *\$channel*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Changes the value of individual pixels based on the intensity of each pixel compared to threshold. The result is a high-contrast, two color image.

Parameters

low

high

channel

Return Values

Returns **TRUE** on success.

Imagick::readImage

Imagick::readImage -- Reads image from filename

Description

bool **Imagick::readImage** (string *\$filename*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Reads image from filename

Parameters

filename

Return Values

Returns **TRUE** on success.

Imagick::readImageBlob

Imagick::readImageBlob -- Reads image from a binary string

Description

bool **Imagick::readImageBlob** (string *\$image* [, string *\$filename*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Reads image from a binary string

Parameters

image

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::readImageFile

Imagick::readImageFile -- Reads image from open filehandle

Description

bool **Imagick::readImageFile** (resource \$filehandle [, string \$fileName])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Reads image from open filehandle

Parameters

filehandle

fileName

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::reduceNoiseImage

Imagick::reduceNoiseImage -- Smooths the contours of an image

Description

bool **Imagick::reduceNoiseImage** (float \$radius)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Smooths the contours of an image while still preserving edge information. The algorithm works by replacing each pixel with its neighbor closest in value. A neighbor is defined by radius. Use a radius of 0 and Imagick::reduceNoiseImage() selects a suitable radius for you.

Parameters

radius

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::removeImage

Imagick::removeImage -- Removes an image from the image list

Description

bool **Imagick::removeImage** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Removes an image from the image list.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::removeImageProfile

Imagick::removeImageProfile -- Removes the named image profile and returns it

Description

string **Imagick::removeImageProfile** (string *\$name*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Removes the named image profile and returns it.

Parameters

name

Return Values

Returns a string containing the profile of the image, throwing ImagickException on error.

Imagick::render

Imagick::render -- Renders all preceding drawing commands

Description

bool **Imagick::render** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Renders all preceding drawing commands.

Return Values

Returns **TRUE** on success.

Imagick::resampleImage

Imagick::resampleImage -- Resample image to desired resolution

Description

bool **Imagick::resampleImage** (float *\$x_resolution*, float *\$y_resolution*, int *\$filter*, float *\$blur*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Resample image to desired resolution.

Parameters

x_resolution

y_resolution

filter

blur

Return Values

Returns **TRUE** on success.

Imagick::resizeImage

Imagick::resizeImage -- Scales an image

Description

```
bool Imagick::resizeImage ( int $columns, int $rows, int $filter, float $blur [, bool $fit ] )
```

Warning

This function is currently not documented; only its argument list is available.

Scales an image to the desired dimensions with a [filter](#).

Parameters

columns

Width of the image

rows

Height of the image

filter

Refer to the list of [filter constants](#).

blur

The blur factor where > 1 is blurry, < 1 is sharp.

fit

Optional fit paramater. Defaults to false.

Return Values

Returns **TRUE** on success.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Added optional fit parameter. This method now supports proportional scaling. Pass |

| | |
|--|--|
| | zero as either parameter for proportional scaling. |
|--|--|

Imagick::rollImage

Imagick::rollImage -- Offsets an image

Description

bool **Imagick::rollImage** (int x , int y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Offsets an image as defined by x and y.

Parameters

x
The X offset.

y
The Y offset.

Return Values

Returns **TRUE** on success.

Imagick::rotateImage

Imagick::rotateImage -- Rotates an image

Description

bool **Imagick::rotateImage** (*mixed* \$background, float \$degrees)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Rotates an image the specified number of degrees. Empty triangles left over from rotating the image are filled with the background color.

Parameters

background

The background color

degrees

The number of degrees to rotate the image

Return Values

Returns **TRUE** on success.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as the first parameter. Previous versions allow only an ImagickPixel object. |

Imagick::roundCorners

Imagick::roundCorners -- Rounds image corners

Description

```
bool Imagick::roundCorners ( float $x_rounding, float $y_rounding [, float $stroke_width [, float $displace [, float $size_correction ]]] )
```

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Rounds image corners. Three last parameters are optional and rarely needed.

Parameters

x_rounding
x rounding

y_rounding
y rounding

stroke_width
stroke width

displace
image displace

size_correction
sise correction

Return Values

Returns **TRUE** on success.

Imagick::sampleImage

Imagick::sampleImage -- Scales an image with pixel sampling

Description

bool **Imagick::sampleImage** (int *\$columns*, int *\$rows*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Scales an image to the desired dimensions with pixel sampling. Unlike other scaling methods, this method does not introduce any additional color into the scaled image.

Parameters

columns

rows

Return Values

Returns **TRUE** on success.

Imagick::scaleImage

Imagick::scaleImage -- Scales the size of an image

Description

bool **Imagick::scaleImage** (int \$cols, int \$rows [, bool \$fit])

Warning

This function is currently not documented; only its argument list is available.

Scales the size of an image to the given dimensions. The other parameter will be calculated if 0 is passed as either param.

Parameters

cols

rows

fit

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|--|
| 2.1.0 | Added optional fit parameter. This method now supports proportional scaling. Pass zero as either parameter for proportional scaling. |

Imagick::separateImageChannel

Imagick::separateImageChannel -- Separates a channel from the image

Description

bool **Imagick::separateImageChannel** (int *\$channel*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Separates a channel from the image and returns a grayscale image. A channel is a particular color component of each pixel in the image.

Parameters

channel

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::sepiaToneImage

Imagick::sepiaToneImage -- Sepia tones an image

Description

bool **Imagick::sepiaToneImage** (float *\$threshold*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Applies a special effect to the image, similar to the effect achieved in a photo darkroom by sepia toning. Threshold ranges from 0 to QuantumRange and is a measure of the extent of the sepia toning. A threshold of 80 is a good starting point for a reasonable tone.

Parameters

threshold

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setBackgroundColor

Imagick::setBackgroundColor -- Sets the object's default background color

Description

bool **Imagick::setBackgroundColor** ([mixed](#) \$background)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the object's default background color.

Parameters

background

Return Values

Returns **TRUE** on success.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as a parameter. Previous versions allow only an ImagickPixel object. |

Imagick::setCompression

Imagick::setCompression -- Sets the object's default compression type

Description

bool **Imagick::setCompression** (int *\$compression*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the object's default compression type

Parameters

compression

Return Values

Returns **TRUE** on success.

Imagick::setCompressionQuality

Imagick::setCompressionQuality -- Sets the object's default compression quality

Description

bool **Imagick::setCompressionQuality** (int *\$quality*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the object's default compression quality.

Parameters

quality

Return Values

Returns **TRUE** on success.

Imagick::setFilename

Imagick::setFilename -- Sets the filename before you read or write the image

Description

bool **Imagick::setFilename** (string *\$filename*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the filename before you read or write an image file.

Parameters

filename

Return Values

Returns **TRUE** on success.

Imagick::setFirstIterator

Imagick::setFirstIterator -- Sets the Imagick iterator to the first image

Description

bool **Imagick::setFirstIterator** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the Imagick iterator to the first image.

Return Values

Returns **TRUE** on success.

Imagick::setFormat

Imagick::setFormat -- Sets the format of the Imagick object

Description

bool **Imagick::setFormat** (string *\$format*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the format of the Imagick object.

Parameters

format

Return Values

Returns **TRUE** on success.

Imagick::setImage

Imagick::setImage -- Replaces image in the object

Description

bool **Imagick::setImage** ([Imagick](#) \$replace)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Replaces the current image sequence with the image from replace object.

Parameters

replace

The replace Imagick object

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Examples

| Example #18 - A Imagick::setImage() example |
|--|
| An example of using Imagick::setImage() <pre><?php /* Create the objects */ \$image = new Imagick('source.jpg'); \$replace = new Imagick('replace.jpg'); /* source.jpg is replaced with replace.jpg */ \$image->setImage(\$replace); /* output the image */ header('Content-type: image/jpeg'); echo \$image;</pre> |

?>

Imagick::setImageBackgroundColor

Imagick::setImageBackgroundColor -- Sets the image background color

Description

bool **Imagick::setImageBackgroundColor** ([mixed](#) \$background)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image background color.

Parameters

background

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as the parameter. Previous versions allow only an ImagickPixel object. |

Imagick::setImageBias

Imagick::setImageBias -- Sets the image bias for any method that convolves an image

Description

bool **Imagick::setImageBias** (float *\$bias*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image bias for any method that convolves an image (e.g. Imagick::ConvolveImage()).

Parameters

bias

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageBluePrimary

Imagick::setImageBluePrimary -- Sets the image chromaticity blue primary point

Description

bool **Imagick::setImageBluePrimary** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image chromaticity blue primary point.

Parameters

x

y

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageBorderColor

Imagick::setImageBorderColor -- Sets the image border color

Description

bool **Imagick::setImageBorderColor** (*mixed* \$border)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image border color.

Parameters

border

The border color

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as a parameter. Previous versions allow only an ImagickPixel object. |

Imagick::setImageChannelDepth

Imagick::setImageChannelDepth -- Sets the depth of a particular image channel

Description

bool **Imagick::setImageChannelDepth** (int *\$channel*, int *\$depth*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the depth of a particular image channel.

Parameters

channel

depth

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageColormapColor

Imagick::setImageColormapColor -- Sets the color of the specified colormap index

Description

bool **Imagick::setImageColormapColor** (int *\$index*, [ImagickPixel](#) *\$color*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the color of the specified colormap index.

Parameters

index

color

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageColorspace

Imagick::setImageColorspace -- Sets the image colorspace

Description

bool **Imagick::setImageColorspace** (int *\$colorspace*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image colorspace.

Parameters

colorspace

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageCompose

Imagick::setImageCompose -- Sets the image composite operator

Description

bool **Imagick::setImageCompose** (int *\$compose*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image composite operator, useful for specifying how to composite the image thumbnail when using the Imagick::montageImage() method.

Parameters

compose

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageCompression

Imagick::setImageCompression -- Sets the image compression

Description

bool **Imagick::setImageCompression** (int *\$compression*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image compression.

Parameters

compression

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageDelay

Imagick::setImageDelay -- Sets the image delay

Description

bool **Imagick::setImageDelay** (int *\$delay*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image delay.

Parameters

delay

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageDepth

Imagick::setImageDepth -- Sets the image depth

Description

bool **Imagick::setImageDepth** (int *\$depth*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image depth.

Parameters

depth

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageDispose

Imagick::setImageDispose -- Sets the image disposal method

Description

bool **Imagick::setImageDispose** (int *\$dispose*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image disposal method.

Parameters

dispose

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageExtent

Imagick::setImageExtent -- Sets the image size

Description

bool **Imagick::setImageExtent** (int *\$columns*, int *\$rows*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image size (i.e. columns & rows).

Parameters

columns

rows

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageFilename

Imagick::setImageFilename -- Sets the filename of a particular image

Description

bool **Imagick::setImageFilename** (string *\$filename*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the filename of a particular image in a sequence.

Parameters

filename

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageFormat

Imagick::setImageFormat -- Sets the format of a particular image

Description

bool **Imagick::setImageFormat** (string *\$format*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the format of a particular image in a sequence.

Parameters

format

String presentation of the image format. Format support depends on the ImageMagick installation.

Return Values

Returns **TRUE** on success.

Imagick::setImageGamma

Imagick::setImageGamma -- Sets the image gamma

Description

bool **Imagick::setImageGamma** (float *\$gamma*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image gamma.

Parameters

gamma

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageGreenPrimary

Imagick::setImageGreenPrimary -- Sets the image chromaticity green primary point

Description

bool **Imagick::setImageGreenPrimary** (float x , float y)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image chromaticity green primary point.

Parameters

x

y

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageIndex

Imagick::setImageIndex -- Set the iterator position

Description

bool **Imagick::setImageIndex** (int *\$index*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Set the iterator to the position in the image list specified with the index parameter.

This method has been deprecated. See [Imagick::setIteratorIndex](#)

Parameters

index

The position to set the iterator to

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageInterlaceScheme

Imagick::setImageInterlaceScheme -- Sets the image compression

Description

bool **Imagick::setImageInterlaceScheme** (int \$interlace_scheme)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image compression.

Parameters

interlace_scheme

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageInterpolateMethod

Imagick::setImageInterpolateMethod -- Sets the image interpolate pixel method

Description

bool **Imagick::setImageInterpolateMethod** (int `$method`)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image interpolate pixel method.

Parameters

method

Return Values

Returns **TRUE** on success.

Imagick::setImageIterations

Imagick::setImageIterations -- Sets the image iterations

Description

bool **Imagick::setImageIterations** (int *\$iterations*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image iterations.

Parameters

iterations

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageMatte

Imagick::setImageMatte -- Sets the image matte channel

Description

bool **Imagick::setImageMatte** (bool *\$matte*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image matte channel.

Parameters

matte

Return Values

Returns **TRUE** on success.

Imagick::setImageMatteColor

Imagick::setImageMatteColor -- Sets the image matte color

Description

bool **Imagick::setImageMatteColor** ([mixed](#) \$matte)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image matte color.

Parameters

matte

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as the parameter. Previous versions allow only an ImagickPixel object. |

Imagick::setImageOpacity

Imagick::setImageOpacity -- Sets the image opacity level

Description

bool **Imagick::setImageOpacity** (float *\$opacity*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image to the specified opacity level. This method is present if Imagick is compiled against ImageMagick 6.3.1 or later.

Parameters

opacity

The level of transparency: 1.0 is fully opaque and 0.0 is fully transparent.

Return Values

Returns **TRUE** on success.

Examples

| Example #19 - A Imagick::setImageOpacity() example |
|---|
| An example of using Imagick::setImageOpacity() <pre><?php /* Create the object */ \$image = new Imagick('source.png'); /* Set the opacity */ \$image->setImageOpacity(0.7); /* output the image */ header('Content-type: image/png'); echo \$image; ?></pre> |

Imagick::setImageOrientation

Imagick::setImageOrientation -- Sets the image orientation.

Description

bool **Imagick::setImageOrientation** (int `$orientation`)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image orientation.

Parameters

orientation

One of the [orientation constants](#)

Return Values

Returns **TRUE** on success.

Imagick::setImagePage

Imagick::setImagePage -- Sets the page geometry of the image

Description

bool **Imagick::setImagePage** (int *\$width*, int *\$height*, int *\$x*, int *\$y*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the page geometry of the image.

Parameters

width

height

x

y

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageProfile

Imagick::setImageProfile -- Adds a named profile to the Imagick object

Description

bool **Imagick::setImageProfile** (string \$name, string \$profile)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds a named profile to the Imagick object. If a profile with the same name already exists, it is replaced. This method differs from the Imagick::ProfileImage() method in that it does not apply any CMS color profiles.

Parameters

name

profile

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageProperty

Imagick::setImageProperty -- Sets an image property

Description

bool **Imagick::setImageProperty** (string *\$name*, string *\$value*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets a named property to the image.

Parameters

name

value

Return Values

Returns **TRUE** on success.

Imagick::setImageRedPrimary

Imagick::setImageRedPrimary -- Sets the image chromaticity red primary point

Description

bool **Imagick::setImageRedPrimary** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image chromaticity red primary point.

Parameters

x

y

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageRenderingIntent

Imagick::setImageRenderingIntent -- Sets the image rendering intent

Description

bool **Imagick::setImageRenderingIntent** (int \$rendering_intent)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image rendering intent.

Parameters

rendering_intent

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageResolution

Imagick::setImageResolution -- Sets the image resolution

Description

bool **Imagick::setImageResolution** (float *\$x_resolution*, float *\$y_resolution*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image resolution.

Parameters

x_resolution

y_resolution

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageScene

Imagick::setImageScene -- Sets the image scene

Description

bool **Imagick::setImageScene** (int *\$scene*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image scene.

Parameters

scene

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setImageTicksPerSecond

Imagick::setImageTicksPerSecond -- Sets the image ticks-per-second

Description

bool **Imagick::setImageTicksPerSecond** (int *\$ticks_per-second*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the image ticks-per-second.

Parameters

ticks_per-second

Return Values

Returns **TRUE** on success.

Imagick::setImageType

Imagick::setImageType -- Sets the image type

Description

bool **Imagick::setImageType** (int \$image_type)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image type.

Parameters

image_type

Return Values

Returns **TRUE** on success.

Imagick::setImageUnits

Imagick::setImageUnits -- Sets the image units of resolution

Description

bool **Imagick::setImageUnits** (int *\$units*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image units of resolution.

Parameters

units

Return Values

Returns **TRUE** on success.

Imagick::setImageVirtualPixelMethod

Imagick::setImageVirtualPixelMethod -- Sets the image virtual pixel method

Description

bool **Imagick::setImageVirtualPixelMethod** (int *\$method*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image virtual pixel method.

Parameters

method

Return Values

Returns **TRUE** on success.

Imagick::setImageWhitePoint

Imagick::setImageWhitePoint -- Sets the image chromaticity white point

Description

bool **Imagick::setImageWhitePoint** (float \$x, float \$y)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image chromaticity white point.

Parameters

x

y

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::setInterlaceScheme

Imagick::setInterlaceScheme -- Sets the image compression

Description

bool **Imagick::setInterlaceScheme** (int *\$interlace_scheme*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image compression.

Parameters

interlace_scheme

Return Values

Returns **TRUE** on success.

Imagick::setIteratorIndex

Imagick::setIteratorIndex -- Set the iterator position

Description

bool **Imagick::setIteratorIndex** (int *\$index*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Set the iterator to the position in the image list specified with the index parameter

Parameters

index

The position to set the iterator to

Return Values

Returns **TRUE** on success.

Imagick::setLastIterator

Imagick::setLastIterator -- Sets the Imagick iterator to the last image

Description

bool **Imagick::setLastIterator** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the Imagick iterator to the last image.

Return Values

Returns **TRUE** on success.

Imagick::setOption

Imagick::setOption -- Set an option

Description

bool **Imagick::setOption** (string *\$key*, string *\$value*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Associates one or options with the wand.

Parameters

key

value

Return Values

Returns **TRUE** on success.

Imagick::setPage

Imagick::setPage -- Sets the page geometry of the Imagick object

Description

bool **Imagick::setPage** (int \$width, int \$height, int \$x, int \$y)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the page geometry of the Imagick object.

Parameters

width

height

x

y

Return Values

Returns **TRUE** on success.

Imagick::setResolution

Imagick::setResolution -- Sets the image resolution

Description

bool **Imagick::setResolution** (float *\$x_resolution*, float *\$y_resolution*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image resolution.

Parameters

x_resolution

y_resolution

Return Values

Returns **TRUE** on success.

Imagick::setResourceLimit

Imagick::setResourceLimit -- Sets the limit for a particular resource in megabytes

Description

bool **Imagick::setResourceLimit** (int *\$type*, int *\$limit*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the limit for a particular resource in megabytes.

Parameters

type

limit

Return Values

Returns **TRUE** on success.

Imagick::setSamplingFactors

Imagick::setSamplingFactors -- Sets the image sampling factors

Description

bool **Imagick::setSamplingFactors** (array *\$factors*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image sampling factors.

Parameters

factors

Return Values

Returns **TRUE** on success.

Imagick::setSize

Imagick::setSize -- Sets the size of the Imagick object

Description

bool **Imagick::setSize** (int *\$columns*, int *\$rows*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the size of the Imagick object. Set it before you read a raw image format such as RGB, GRAY, or CMYK.

Parameters

columns

rows

Return Values

Returns **TRUE** on success.

Imagick::setSizeOffset

Imagick::setSizeOffset -- Sets the size and offset of the Imagick object

Description

bool **Imagick::setSizeOffset** (int *\$columns*, int *\$rows*, int *\$offset*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the size and offset of the Imagick object. Set it before you read a raw image format such as RGB, GRAY, or CMYK.

Parameters

columns

rows

offset

Return Values

Returns **TRUE** on success.

Imagick::setType

Imagick::setType -- Sets the image type attribute

Description

bool **Imagick::setType** (int \$image_type)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the image type attribute.

Parameters

image_type

Return Values

Returns **TRUE** on success.

Imagick::shadedImage

Imagick::shadedImage -- Creates a 3D effect

Description

bool **Imagick::shadedImage** (bool \$gray, float \$azimuth, float \$elevation)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Shines a distant light on an image to create a three-dimensional effect. You control the positioning of the light with azimuth and elevation; azimuth is measured in degrees off the x axis and elevation is measured in pixels above the Z axis.

Parameters

gray

azimuth

elevation

Return Values

Returns **TRUE** on success.

Imagick::shadowImage

Imagick::shadowImage -- Simulates an image shadow

Description

bool **Imagick::shadowImage** (float *\$opacity*, float *\$sigma*, int *\$x*, int *\$y*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Simulates an image shadow.

Parameters

opacity

sigma

x

y

Return Values

Returns **TRUE** on success.

Imagick::sharpenImage

Imagick::sharpenImage -- Sharpens an image

Description

bool **Imagick::sharpenImage** (float \$radius, float \$sigma [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sharpens an image. We convolve the image with a Gaussian operator of the given radius and standard deviation (sigma). For reasonable results, the radius should be larger than sigma. Use a radius of 0 and selects a suitable radius for you.

Parameters

radius

sigma

channel

Return Values

Returns **TRUE** on success.

Imagick::shaveImage

Imagick::shaveImage -- Shaves pixels from the image edges

Description

bool **Imagick::shaveImage** (int `$columns`, int `$rows`)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Shaves pixels from the image edges. It allocates the memory necessary for the new Image structure and returns a pointer to the new image.

Parameters

columns

rows

Return Values

Returns **TRUE** on success.

Imagick::shearImage

Imagick::shearImage -- Creating a parallelogram

Description

bool **Imagick::shearImage** ([mixed](#) \$background, float \$x_shear, float \$y_shear)

Warning

This function is currently not documented; only its argument list is available.

Slides one edge of an image along the X or Y axis, creating a parallelogram. An X direction shear slides an edge along the X axis, while a Y direction shear slides an edge along the Y axis. The amount of the shear is controlled by a shear angle. For X direction shears, x_shear is measured relative to the Y axis, and similarly, for Y direction shears y_shear is measured relative to the X axis. Empty triangles left over from shearing the image are filled with the background color.

Parameters

background

The background color

x_shear

The number of degrees to shear on the x axis

y_shear

The number of degrees to shear on the y axis

Return Values

Returns **TRUE** on success.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as the first parameter. Previous versions allow only an ImagickPixel object. |

Imagick::sigmoidalContrastImage

Imagick::sigmoidalContrastImage -- Adjusts the contrast of an image

Description

```
bool Imagick::sigmoidalContrastImage ( bool $sharpen, float $alpha, float $beta [, int $channel ] )
```

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adjusts the contrast of an image with a non-linear sigmoidal contrast algorithm. Increase the contrast of the image using a sigmoidal transfer function without saturating highlights or shadows. Contrast indicates how much to increase the contrast (0 is none; 3 is typical; 20 is pushing it); mid-point indicates where midtones fall in the resultant image (0 is white; 50 is middle-gray; 100 is black). Set sharpen to **TRUE** to increase the image contrast otherwise the contrast is reduced.

Parameters

sharpen

alpha

beta

channel

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::sketchImage

Imagick::sketchImage -- Simulates a pencil sketch

Description

bool **Imagick::sketchImage** (float *\$radius*, float *\$sigma*, float *\$angle*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Simulates a pencil sketch. We convolve the image with a Gaussian operator of the given radius and standard deviation (sigma). For reasonable results, radius should be larger than sigma. Use a radius of 0 and Imagick::sketchImage() selects a suitable radius for you. Angle gives the angle of the blurring motion.

Parameters

radius

sigma

angle

Return Values

Returns **TRUE** on success.

Imagick::solarizeImage

Imagick::solarizeImage -- Applies a solarizing effect to the image

Description

bool **Imagick::solarizeImage** (int *\$threshold*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Applies a special effect to the image, similar to the effect achieved in a photo darkroom by selectively exposing areas of photo sensitive paper to light. Threshold ranges from 0 to QuantumRange and is a measure of the extent of the solarization.

Parameters

threshold

Return Values

Returns **TRUE** on success.

Imagick::spliceImage

Imagick::spliceImage -- Splices a solid color into the image

Description

bool **Imagick::spliceImage** (int \$width, int \$height, int \$x, int \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Splices a solid color into the image.

Parameters

width

height

x

y

Return Values

Returns **TRUE** on success.

Imagick::spreadImage

Imagick::spreadImage -- Randomly displaces each pixel in a block

Description

bool **Imagick::spreadImage** (float \$radius)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Special effects method that randomly displaces each pixel in a block defined by the radius parameter.

Parameters

radius

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::steganolImage

Imagick::steganolImage -- Hides a digital watermark within the image

Description

[Imagick](#) Imagick::steganolImage ([Imagick](#) \$watermark_wand, int \$offset)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Hides a digital watermark within the image. Recover the hidden watermark later to prove that the authenticity of an image. Offset defines the start position within the image to hide the watermark.

Parameters

watermark_wand

offset

Return Values

Returns **TRUE** on success.

Imagick::stereolmage

Imagick::stereolmage -- Composites two images

Description

bool **Imagick::stereolmage** ([Imagick](#) \$offset_wand)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Composites two images and produces a single image that is the composite of a left and right image of a stereo pair.

Parameters

offset_wand

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::stripImage

Imagick::stripImage -- Strips an image of all profiles and comments

Description

bool **Imagick::stripImage** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Strips an image of all profiles and comments.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::swirlImage

Imagick::swirlImage -- Swirls the pixels about the center of the image

Description

bool **Imagick::swirlImage** (float \$degrees)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Swirls the pixels about the center of the image, where degrees indicates the sweep of the arc through which each pixel is moved. You get a more dramatic effect as the degrees move from 1 to 360.

Parameters

degrees

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::textureImage

Imagick::textureImage -- Repeatedly tiles the texture image

Description

bool **Imagick::textureImage** ([Imagick](#) \$texture_wand)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Repeatedly tiles the texture image across and down the image canvas.

Parameters

texture_wand

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::thresholdImage

Imagick::thresholdImage -- Changes the value of individual pixels based on a threshold

Description

bool **Imagick::thresholdImage** (float *\$threshold* [, int *\$channel*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Changes the value of individual pixels based on the intensity of each pixel compared to threshold. The result is a high-contrast, two color image.

Parameters

threshold

channel

Return Values

Returns **TRUE** on success.

Imagick::thumbnailImage

Imagick::thumbnailImage -- Changes the size of an image

Description

bool **Imagick::thumbnailImage** (int *\$columns*, int *\$rows* [, bool *\$fit*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Changes the size of an image to the given dimensions and removes any associated profiles. The goal is to produce small low cost thumbnail images suited for display on the Web. If **TRUE** is given as a third parameter then columns and rows parameters are used as maximums for each side. Both sides will be scaled down until the match or are smaller than the parameter given for the side.

Parameters

columns

Image width

rows

Image height

fit

Whether to force maximum values

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::tintImage

Imagick::tintImage -- Applies a color vector to each pixel in the image

Description

bool **Imagick::tintImage** (*mixed* \$tint, *mixed* \$opacity)

Warning

This function is currently not documented; only its argument list is available.

Applies a color vector to each pixel in the image. The length of the vector is 0 for black and white and at its maximum for the midtones. The vector weighing function is $f(x)=(1-(4.0*((x-0.5)*(x-0.5))))$.

Parameters

tint

opacity

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as the first parameter and a float representing the opacity value as the second parameter. Previous versions allow only an ImagickPixel objects. |

Imagick::transformImage

Imagick::transformImage -- Convenience method for setting crop size and the image geometry

Description

[Imagick](#) **Imagick::transformImage** (string *\$crop*, string *\$geometry*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

A convenience method for setting crop size and the image geometry from strings

Parameters

crop

A crop geometry string. This geometry defines a subregion of the image to crop.

geometry

An image geometry string. This geometry defines the final size of the image.

Return Values

Returns **TRUE** on success.

Examples

| Example #20 - Using Imagick::transformImage() : |
|---|
| The example creates a 100x100 black image. <pre><?php \$image = new Imagick(); \$image->newImage(300, 200, "black"); \$new_image = \$image->transformImage("100x100", "100x100"); \$new_image->writeImage('test_out.jpg'); ?></pre> |

See Also

- [Imagick::adaptiveBlurImage\(\)](#)
- [Imagick::motionBlurImage\(\)](#)
- [Imagick::radialBlurImage\(\)](#)

Imagick::transverseImage

Imagick::transverseImage -- Creates a horizontal mirror image

Description

bool **Imagick::transverseImage** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Creates a horizontal mirror image by reflecting the pixels around the central y-axis while rotating them 270-degrees.

Return Values

Returns **TRUE** on success.

Imagick::trimImage

Imagick::trimImage -- Remove edges from the image

Description

bool **Imagick::trimImage** (float *\$fuzz*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Remove edges that are the background color from the image.

Parameters

fuzz

By default target must match a particular pixel color exactly. However, in many cases two colors may differ by a small amount. The fuzz member of image defines how much tolerance is acceptable to consider two colors as the same.

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Examples

| |
|---|
| Example #21 - Using Imagick::trimImage(): |
| Trim an image, then display to the browser. <pre><?php /* Create the object and read the image in */ \$im = new Imagick("image.jpg"); /* Trim the image. */ \$im->trimImage(0); /* Output the image */</pre> |


```
header( "Content-Type: image/" . $im->getImageFormat() );  
echo $im;  
?>
```

Imagick::uniqueImageColors

Imagick::uniqueImageColors -- Discards all but one of any pixel color

Description

bool **Imagick::uniqueImageColors** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Discards all but one of any pixel color.

Return Values

Returns **TRUE** on success.

Imagick::unsharpMaskImage

Imagick::unsharpMaskImage -- Sharpens an image

Description

bool **Imagick::unsharpMaskImage** (float \$radius, float \$sigma, float \$amount, float \$threshold [, int \$channel])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sharpens an image. We convolve the image with a Gaussian operator of the given radius and standard deviation (sigma). For reasonable results, radius should be larger than sigma. Use a radius of 0 and Imagick::UnsharpMaskImage() selects a suitable radius for you.

Parameters

radius

sigma

amount

threshold

channel

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::valid

Imagick::valid -- Checks if the current item is valid

Description

bool **Imagick::valid** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Checks if the current item is valid.

Return Values

Returns **TRUE** on success.

Imagick::vignetteImage

Imagick::vignetteImage -- Adds vignette filter to the image

Description

bool **Imagick::vignetteImage** (float \$blackPoint, float \$whitePoint, int \$x, int \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds vignette filter to the image.

Parameters

blackPoint

whitePoint

x

y

Return Values

Returns **TRUE** on success.

Imagick::waveImage

Imagick::waveImage -- Adds wave filter to the image

Description

bool **Imagick::waveImage** (float *\$amplitude*, float *\$length*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds wave filter to the image.

Parameters

amplitude

length

Return Values

Returns **TRUE** on success.

Errors/Exceptions

Throws ImagickException on error.

Imagick::whiteThresholdImage

Imagick::whiteThresholdImage -- Force all pixels above the threshold into white

Description

bool Imagick::whiteThresholdImage (mixed \$threshold)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Is like Imagick::ThresholdImage() but force all pixels above the threshold into white while leaving all pixels below the threshold unchanged.

Parameters

threshold

Return Values

Returns **TRUE** on success.

ChangeLog

| Version | Description |
|---------|---|
| 2.1.0 | Now allows a string representing the color as a parameter. Previous versions allow only an ImagickPixel object. |

Imagick::writeImage

Imagick::writeImage -- Writes an image to the specified filename

Description

bool **Imagick::writeImage** ([string *\$filename*])

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Writes an image to the specified filename. If the filename parameter is NULL, the image is written to the filename set by Imagick::ReadImage() or Imagick::SetImageFilename().

Parameters

filename

Return Values

Returns **TRUE** on success.

Imagick::writeImages

Imagick::writeImages -- Writes an image or image sequence

Description

bool **Imagick::writeImages** (string *\$filename*, bool *\$adjoin*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Writes an image or image sequence.

Parameters

filename

adjoin

Return Values

Returns **TRUE** on success.

The ImagicDraw class

Class synopsis

| |
|--------------------|
| ImagickDraw |
|--------------------|

```
ImagickDraw {  
  
    bool ImagickDraw::affine ( array $affine )  
  
    bool ImagickDraw::annotation ( float $x, float $y, string $text )  
  
    bool ImagickDraw::arc ( float $sx, float $sy, float $ex, float $ey, float $sd, float $ed )  
  
    bool ImagickDraw::bezier ( array $coordinates )  
  
    bool ImagickDraw::circle ( float $ox, float $oy, float $px, float $py )  
  
    bool ImagickDraw::clear ( void )  
  
    ImagickDraw ImagickDraw::clone ( void )  
  
    bool ImagickDraw::color ( float $x, float $y, int $paintMethod )  
  
    bool ImagickDraw::comment ( string $comment )  
  
    bool ImagickDraw::composite ( int $compose, float $x, float $y, float $width, float $  
height, Imagick $compositeWand )  
  
    ImagickDraw ImagickDraw::__construct ( void )  
  
    bool ImagickDraw::destroy ( void )  
  
    bool ImagickDraw::ellipse ( float $ox, float $oy, float $rx, float $ry, float $start,  
float $end )  
  
    string ImagickDraw::getClipPath ( void )  
  
    int ImagickDraw::getClipRule ( void )  
  
    int ImagickDraw::getClipUnits ( void )  
  
    ImagickPixel ImagickDraw::getFillColor ( void )
```

float ImagickDraw::getFillOpacity (void)

int ImagickDraw::getFillRule (void)

string ImagickDraw::getFont (void)

string ImagickDraw::getFontFamily (void)

float ImagickDraw::getFontSize (void)

int ImagickDraw::getFontStyle (void)

int ImagickDraw::getFontWeight (void)

int ImagickDraw::getGravity (void)

bool ImagickDraw::getStrokeAntialias (void)

ImagickPixel ImagickDraw::getStrokeColor ([ImagickPixel](#) \$stroke_color)

array ImagickDraw::getStrokeDashArray (void)

float ImagickDraw::getStrokeDashOffset (void)

int ImagickDraw::getStrokeLineCap (void)

int ImagickDraw::getStrokeLineJoin (void)

int ImagickDraw::getStrokeMiterLimit (void)

float ImagickDraw::getStrokeOpacity (void)

float ImagickDraw::getStrokeWidth (void)

int ImagickDraw::getTextAlignment (void)

bool ImagickDraw::getTextAntialias (void)

int ImagickDraw::getTextDecoration (void)

string ImagickDraw::getTextEncoding (void)

ImagickPixel ImagickDraw::getTextUnderColor (void)

string ImagickDraw::getVectorGraphics (void)

bool ImagickDraw::line (float \$sx, float \$sy, float \$ex, float \$ey)

bool ImagickDraw::matte (float \$x, float \$y, int \$paintMethod)

bool ImagickDraw::pathClose (void)

bool MagickDraw::pathCurveToAbsolute (float \$x1, float \$y1, float \$x2, float \$y2, float \$x, float \$y)

bool MagickDraw::pathCurveToQuadraticBezierAbsolute (float \$x1, float \$y1, float \$x, float \$y)

bool MagickDraw::pathCurveToQuadraticBezierRelative (float \$x1, float \$y1, float \$x, float \$y)

bool MagickDraw::pathCurveToQuadraticBezierSmoothAbsolute (float \$x, float \$y)

bool MagickDraw::pathCurveToQuadraticBezierSmoothRelative (float \$x, float \$y)

bool MagickDraw::pathCurveToRelative (float \$x1, float \$y1, float \$x2, float \$y2, float \$x, float \$y)

bool MagickDraw::pathCurveToSmoothAbsolute (float \$x2, float \$y2, float \$x, float \$y)

bool MagickDraw::pathCurveToSmoothRelative (float \$x2, float \$y2, float \$x, float \$y)

bool MagickDraw::pathEllipticArcAbsolute (float \$rx, float \$ry, float \$x_axis_rotation, bool \$large_arc_flag, bool \$sweep_flag, float \$x, float \$y)

bool MagickDraw::pathEllipticArcRelative (float \$rx, float \$ry, float \$x_axis_rotation, bool \$large_arc_flag, bool \$sweep_flag, float \$x, float \$y)

bool MagickDraw::pathFinish (void)

bool MagickDraw::pathLineToAbsolute (float \$x, float \$y)

bool MagickDraw::pathLineToHorizontalAbsolute (float \$x)

bool MagickDraw::pathLineToHorizontalRelative (float \$x)

bool MagickDraw::pathLineToRelative (float \$x, float \$y)

bool MagickDraw::pathLineToVerticalAbsolute (float \$y)

bool MagickDraw::pathLineToVerticalRelative (float \$y)

bool MagickDraw::pathMoveToAbsolute (float \$x, float \$y)

bool MagickDraw::pathMoveToRelative (float \$x, float \$y)

bool MagickDraw::pathStart (void)

bool MagickDraw::point (float \$x, float \$y)

bool MagickDraw::polygon (array \$coordinates)

bool ImagickDraw::polyline (array \$coordinates)

bool ImagickDraw::pop (void)

bool ImagickDraw::popClipPath (void)

bool ImagickDraw::popDefs (void)

bool ImagickDraw::popPattern (void)

bool ImagickDraw::push (void)

bool ImagickDraw::pushClipPath (string \$clip_mask_id)

bool ImagickDraw::pushDefs (void)

bool ImagickDraw::pushPattern (string \$pattern_id, float \$x, float \$y, float \$width, float \$height)

bool ImagickDraw::rectangle (float \$x1, float \$y1, float \$x2, float \$y2)

bool ImagickDraw::render (void)

bool ImagickDraw::rotate (float \$degrees)

bool ImagickDraw::roundRectangle (float \$x1, float \$y1, float \$x2, float \$y2, float \$rx, float \$ry)

bool ImagickDraw::scale (float \$x, float \$y)

bool ImagickDraw::setClipPath (string \$clip_mask)

bool ImagickDraw::setClipRule (int \$fill_rule)

bool ImagickDraw::setClipUnits (int \$clip_units)

bool ImagickDraw::setFillAlpha (float \$opacity)

bool ImagickDraw::setFillColor ([ImagickPixel](#) \$fill_pixel)

bool ImagickDraw::setFillOpacity (float \$fillOpacity)

bool ImagickDraw::setFillPatternURL (string \$fill_url)

bool ImagickDraw::setFillRule (int \$fill_rule)

bool ImagickDraw::setFont (string \$font_name)

bool ImagickDraw::setFontFamily (string \$font_family)

bool ImagickDraw::setFontSize (float \$pointsize)

```
bool ImagickDraw::setFontStretch ( int $fontStretch )

bool ImagickDraw::setFontStyle ( int $style )

bool ImagickDraw::setFontWeight ( int $font_weight )

bool ImagickDraw::setGravity ( int $gravity )

bool ImagickDraw::setStrokeAlpha ( float $opacity )

bool ImagickDraw::setStrokeAntialias ( bool $stroke_antialias )

bool ImagickDraw::setStrokeColor ( ImagickPixel $stroke_pixel )

bool ImagickDraw::setStrokeDashArray ( array $dashArray )

bool ImagickDraw::setStrokeDashOffset ( float $dash_offset )

bool ImagickDraw::setStrokeLineCap ( int $linecap )

bool ImagickDraw::setStrokeLineJoin ( int $linejoin )

bool ImagickDraw::setStrokeMiterLimit ( int $miterlimit )

bool ImagickDraw::setStrokeOpacity ( float $stroke_opacity )

bool ImagickDraw::setStrokePatternURL ( string $stroke_url )

bool ImagickDraw::setStrokeWidth ( float $stroke_width )

bool ImagickDraw::setTextAlignment ( int $alignment )

bool ImagickDraw::setTextAntialias ( bool $antiAlias )

bool ImagickDraw::setTextDecoration ( int $decoration )

bool ImagickDraw::setTextEncoding ( string $encoding )

bool ImagickDraw::setTextUnderColor ( ImagickPixel $under_color )

bool ImagickDraw::setVectorGraphics ( string $xml )

bool ImagickDraw::setViewbox ( int $x1, int $y1, int $x2, int $y2 )

bool ImagickDraw::skewX ( float $degrees )

bool ImagickDraw::skewY ( float $degrees )

bool ImagickDraw::translate ( float $x, float $y )
}
```

ImagickDraw::affine

ImagickDraw::affine -- Adjusts the current affine transformation matrix

Description

bool **ImagickDraw::affine** (array *\$affine*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Adjusts the current affine transformation matrix with the specified affine transformation matrix.

Parameters

affine

Affine matrix parameters

Return Values

No value is returned.

ImagickDraw::annotation

ImagickDraw::annotation -- Draws text on the image

Description

bool **ImagickDraw::annotation** (float *\$x*, float *\$y*, string *\$text*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws text on the image.

Parameters

x
The x coordinate where text is drawn

y
The y coordinate where text is drawn

text
The text to draw on the image

Return Values

No value is returned.

ImagickDraw::arc

ImagickDraw::arc -- Draws an arc

Description

bool **ImagickDraw::arc** (float *\$sx*, float *\$sy*, float *\$ex*, float *\$ey*, float *\$sd*, float *\$ed*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws an arc falling within a specified bounding rectangle on the image.

Parameters

sx
Starting x ordinate of bounding rectangle

sy
starting y ordinate of bounding rectangle

ex
ending x ordinate of bounding rectangle

ey
ending y ordinate of bounding rectangle

sd
starting degrees of rotation

ed
ending degrees of rotation

Return Values

No value is returned.

ImagickDraw::bezier

ImagickDraw::bezier -- Draws a bezier curve

Description

bool **ImagickDraw::bezier** (array \$coordinates)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Draws a bezier curve through a set of points on the image.

Parameters

coordinates

Multidimensional array like array(array('x' => 1, 'y' => 2), array('x' => 3, 'y' => 4))

Return Values

No value is returned.

ImagickDraw::circle

ImagickDraw::circle -- Draws a circle

Description

bool **ImagickDraw::circle** (float \$ox, float \$oy, float \$px, float \$py)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a circle on the image.

Parameters

ox
origin x coordinate

oy
origin y coordinate

px
perimeter x coordinate

py
perimeter y coordinate

Return Values

No value is returned.

ImagickDraw::clear

ImagickDraw::clear -- Clears the ImagickDraw

Description

bool **ImagickDraw::clear** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Clears the ImagickDraw object of any accumulated commands, and resets the settings it contains to their defaults.

Return Values

Returns an ImagickDraw object.

ImagickDraw::clone

ImagickDraw::clone -- Makes an exact copy of the specified ImagickDraw object

Description

[ImagickDraw](#) **ImagickDraw::clone** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Makes an exact copy of the specified ImagickDraw object.

Return Values

What the function returns, first on success, then on failure. See also the &return.success; entity

ImagickDraw::color

ImagickDraw::color -- Draws color on image

Description

bool **ImagickDraw::color** (float *\$x*, float *\$y*, int *\$paintMethod*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws color on image using the current fill color, starting at specified position, and using specified paint method.

Parameters

x
x coordinate of the paint

y
y coordinate of the paint

paintMethod
one of the PAINT_ constants

Return Values

No value is returned.

ImagickDraw::comment

ImagickDraw::comment -- Adds a comment

Description

bool **ImagickDraw::comment** (string *\$comment*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adds a comment to a vector output stream.

Parameters

comment

The comment string to add to vector output stream

Return Values

No value is returned.

ImagickDraw::composite

ImagickDraw::composite -- Composites an image onto the current image

Description

```
bool ImagickDraw::composite ( int $compose, float $x, float $y, float $width, float $height, Imagick $compositeWand )
```

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Composites an image onto the current image, using the specified composition operator, specified position, and at the specified size.

Parameters

compose
composition operator. One of COMPOSITE_ constants

x
x coordinate of the top left corner

y
y coordinate of the top left corner

width
width of the composition image

height
height of the composition image

compositeWand
the Imagick object where composition image is taken from

Return Values

Returns **TRUE** on success.

ImagickDraw::__construct

ImagickDraw::__construct -- The ImagickDraw constructor

Description

[ImagickDraw](#) **ImagickDraw::__construct** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

The ImagickDraw constructor

Return Values

No value is returned.

ImagickDraw::destroy

ImagickDraw::destroy -- Frees all associated resources

Description

bool **ImagickDraw::destroy** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Frees all resources associated with the ImagickDraw object.

Return Values

No value is returned.

ImagickDraw::ellipse

ImagickDraw::ellipse -- Draws an ellipse on the image

Description

```
bool ImagickDraw::ellipse ( float $ox, float $oy, float $rx, float $ry, float $start, float $end )
```

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws an ellipse on the image.

Parameters

ox

oy

rx

ry

start

end

Return Values

No value is returned.

ImagickDraw::getClipPath

ImagickDraw::getClipPath -- Obtains the current clipping path ID

Description

string **ImagickDraw::getClipPath** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Obtains the current clipping path ID.

Return Values

Returns a string containing the clip path ID or false if no clip path exists.

ImagickDraw::getClipRule

ImagickDraw::getClipRule -- Returns the current polygon fill rule

Description

int **ImagickDraw::getClipRule** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the current polygon fill rule to be used by the clipping path.

Return Values

Returns one of the FILLRULE_ constants.

ImagickDraw::getClipUnits

ImagickDraw::getClipUnits -- Returns the interpretation of clip path units

Description

int **ImagickDraw::getClipUnits** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the interpretation of clip path units.

Return Values

Returns an int on success.

ImagickDraw::getFillColor

ImagickDraw::getFillColor -- Returns the fill color

Description

[ImagickPixel](#) ImagickDraw::getFillColor (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the fill color used for drawing filled objects.

Return Values

Returns an ImagickPixel object.

ImagickDraw::getFillOpacity

ImagickDraw::getFillOpacity -- Returns the opacity used when drawing

Description

float **ImagickDraw::getFillOpacity** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the opacity used when drawing using the fill color or fill texture. Fully opaque is 1.0.

Return Values

What the function returns, first on success, then on failure. See also the &return.success; entity

ImagickDraw::getFillRule

ImagickDraw::getFillRule -- Returns the fill rule

Description

int **ImagickDraw::getFillRule** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the fill rule used while drawing polygons.

Return Values

Returns a FILLRULE_ constant

ImagickDraw::getFont

ImagickDraw::getFont -- Returns the font

Description

string **ImagickDraw::getFont** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns a string specifying the font used when annotating with text.

Return Values

Returns a string on success and false if no font is set.

ImagickDraw::getFontFamily

ImagickDraw::getFontFamily -- Returns the font family

Description

string **ImagickDraw::getFontFamily** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the font family to use when annotating with text.

Return Values

Returns the font family currently selected or false if font family is not set.

ImagickDraw::getFontSize

ImagickDraw::getFontSize -- Returns the font pointsize

Description

float **ImagickDraw::getFontSize** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the font pointsize used when annotating with text.

Return Values

Returns the font size associated with the current ImagickDraw object.

ImagickDraw::getFontStyle

ImagickDraw::getFontStyle -- Returns the font style

Description

int **ImagickDraw::getFontStyle** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the font style used when annotating with text.

Return Values

Returns the font style constant (STYLE_) associated with the ImagickDraw object or 0 if no style is set.

ImagickDraw::getFontWeight

ImagickDraw::getFontWeight -- Returns the font weight

Description

int **ImagickDraw::getFontWeight** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the font weight used when annotating with text.

Return Values

Returns an int on success and 0 if no weight is set.

ImagickDraw::getGravity

ImagickDraw::getGravity -- Returns the text placement gravity

Description

int **ImagickDraw::getGravity** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the text placement gravity used when annotating with text.

Return Values

Returns a GRAVITY_ constant on success and 0 if no gravity is set.

ImagickDraw::getStrokeAntialias

ImagickDraw::getStrokeAntialias -- Returns the current stroke antialias setting

Description

bool **ImagickDraw::getStrokeAntialias** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the current stroke antialias setting. Stroked outlines are antialiased by default. When antialiasing is disabled stroked pixels are thresholded to determine if the stroke color or underlying canvas color should be used.

Return Values

Returns **TRUE** if antialiasing is on and false if it is off.

ImagickDraw::getStrokeColor

ImagickDraw::getStrokeColor -- Returns the color used for stroking object outlines

Description

[ImagickPixel](#) ImagickDraw::getStrokeColor ([ImagickPixel](#) \$stroke_color)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the color used for stroking object outlines.

Parameters

stroke_color

Return Values

Returns an ImagickPixel object which describes the color.

ImagickDraw::getStrokeDashArray

ImagickDraw::getStrokeDashArray -- Returns an array representing the pattern of dashes and gaps used to stroke paths

Description

array **ImagickDraw::getStrokeDashArray** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns an array representing the pattern of dashes and gaps used to stroke paths.

Return Values

Returns an array on success and empty array if not set.

ImagickDraw::getStrokeDashOffset

ImagickDraw::getStrokeDashOffset -- Returns the offset into the dash pattern to start the dash

Description

float **ImagickDraw::getStrokeDashOffset** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the offset into the dash pattern to start the dash.

Return Values

Returns a float representing the offset and 0 if it's not set.

ImagickDraw::getStrokeLineCap

ImagickDraw::getStrokeLineCap -- Returns the shape to be used at the end of open subpaths when they are stroked

Description

int **ImagickDraw::getStrokeLineCap** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the shape to be used at the end of open subpaths when they are stroked.

Return Values

Returns one of the LINECAP_ constants or 0 if stroke linecap is not set.

ImagickDraw::getStrokeLineJoin

ImagickDraw::getStrokeLineJoin -- Returns the shape to be used at the corners of paths when they are stroked

Description

int **ImagickDraw::getStrokeLineJoin** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the shape to be used at the corners of paths (or other vector shapes) when they are stroked.

Return Values

Returns one of the LINEJOIN_ constants or 0 if stroke line join is not set.

ImagickDraw::getStrokeMiterLimit

ImagickDraw::getStrokeMiterLimit -- Returns the stroke miter limit

Description

int **ImagickDraw::getStrokeMiterLimit** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the miter limit. When two line segments meet at a sharp angle and miter joins have been specified for 'lineJoin', it is possible for the miter to extend far beyond the thickness of the line stroking the path. The miterLimit' imposes a limit on the ratio of the miter length to the 'lineWidth'.

Return Values

Returns an int describing the miter limit and 0 if no miter limit is set.

ImagickDraw::getStrokeOpacity

ImagickDraw::getStrokeOpacity -- Returns the opacity of stroked object outlines

Description

float **ImagickDraw::getStrokeOpacity** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the opacity of stroked object outlines.

Return Values

Returns a double describing the opacity.

ImagickDraw::getStrokeWidth

ImagickDraw::getStrokeWidth -- Returns the width of the stroke used to draw object outlines

Description

float **ImagickDraw::getStrokeWidth** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the width of the stroke used to draw object outlines.

Return Values

Returns a double describing the stroke width.

ImagickDraw::getTextAlignment

ImagickDraw::getTextAlignment -- Returns the text alignment

Description

int **ImagickDraw::getTextAlignment** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the alignment applied when annotating with text.

Return Values

Returns one of the ALIGN_ constants and 0 if no align is set.

ImagickDraw::getTextAntialias

ImagickDraw::getTextAntialias -- Returns the current text antialias setting

Description

bool **ImagickDraw::getTextAntialias** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the current text antialias setting, which determines whether text is antialiased. Text is antialiased by default.

Return Values

Returns **TRUE** if text is antialiased and false if not.

ImagickDraw::getTextDecoration

ImagickDraw::getTextDecoration -- Returns the text decoration

Description

int **ImagickDraw::getTextDecoration** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the decoration applied when annotating with text.

Return Values

Returns one of the DECORATION_ constants and 0 if no decoration is set.

ImagickDraw::getTextEncoding

ImagickDraw::getTextEncoding -- Returns the code set used for text annotations

Description

string **ImagickDraw::getTextEncoding** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns a string which specifies the code set used for text annotations.

Return Values

Returns a string specifying the code set or false if text encoding is not set.

ImagickDraw::getTextUnderColor

ImagickDraw::getTextUnderColor -- Returns the text under color

Description

[ImagickPixel](#) ImagickDraw::getTextUnderColor (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the color of a background rectangle to place under text annotations.

Return Values

Returns an ImagickPixel object describing the color.

ImagickDraw::getVectorGraphics

ImagickDraw::getVectorGraphics -- Returns a string containing vector graphics

Description

string **ImagickDraw::getVectorGraphics** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns a string which specifies the vector graphics generated by any graphics calls made since the ImagickDraw object was instantiated.

Return Values

Returns a string containing the vector graphics.

ImagickDraw::line

ImagickDraw::line -- Draws a line

Description

bool **ImagickDraw::line** (float *\$sx*, float *\$sy*, float *\$ex*, float *\$ey*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a line on the image using the current stroke color, stroke opacity, and stroke width.

Parameters

sx
starting x coordinate

sy
starting y coordinate

ex
ending x coordinate

ey
ending y coordinate

Return Values

No value is returned.

ImagickDraw::matte

ImagickDraw::matte -- Paints on the image's opacity channel

Description

bool **ImagickDraw::matte** (float \$x, float \$y, int \$paintMethod)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Paints on the image's opacity channel in order to set effected pixels to transparent. to influence the opacity of pixels.

Parameters

x
x coordinate of the matte

y
y coordinate of the matte

paintMethod
PAINT_ constant

Return Values

What the function returns, first on success, then on failure. See also the &return.success; entity

ImagickDraw::pathClose

ImagickDraw::pathClose -- Adds a path element to the current path

Description

bool **ImagickDraw::pathClose** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Adds a path element to the current path which closes the current subpath by drawing a straight line from the current point to the current subpath's most recent starting point (usually, the most recent moveto point).

Return Values

No value is returned.

ImagickDraw::pathCurveToAbsolute

ImagickDraw::pathCurveToAbsolute -- Draws a cubic Bezier curve

Description

bool **ImagickDraw::pathCurveToAbsolute** (float \$x1, float \$y1, float \$x2, float \$y2, float \$x, float \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a cubic Bezier curve from the current point to (x,y) using (x1,y1) as the control point at the beginning of the curve and (x2,y2) as the control point at the end of the curve using absolute coordinates. At the end of the command, the new current point becomes the final (x,y) coordinate pair used in the polybezier.

Parameters

x1
x coordinate of the first control point

y1
y coordinate of the first control point

x2
x coordinate of the second control point

y2
y coordinate of the first control point

x
x coordinate of the curve end

y
y coordinate of the curve end

Return Values

No value is returned.

ImagickDraw::pathCurveToQuadraticBezierAbsolute

ImagickDraw::pathCurveToQuadraticBezierAbsolute -- Draws a quadratic Bezier curve

Description

bool **ImagickDraw::pathCurveToQuadraticBezierAbsolute** (float x_1 , float y_1 , float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a quadratic Bezier curve from the current point to (x,y) using (x1,y1) as the control point using absolute coordinates. At the end of the command, the new current point becomes the final (x,y) coordinate pair used in the polybezier.

Parameters

x_1
x coordinate of the control point

y_1
y coordinate of the control point

x
x coordinate of the end point

y
y coordinate of the end point

Return Values

No value is returned.

ImagickDraw::pathCurveToQuadraticBezierRelative

ImagickDraw::pathCurveToQuadraticBezierRelative -- Draws a quadratic Bezier curve

Description

bool **ImagickDraw::pathCurveToQuadraticBezierRelative** (float x_1 , float y_1 , float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a quadratic Bezier curve from the current point to (x,y) using (x1,y1) as the control point using relative coordinates. At the end of the command, the new current point becomes the final (x,y) coordinate pair used in the polybezier.

Parameters

x_1
starting x coordinate

y_1
starting y coordinate

x
ending x coordinate

y
ending y coordinate

Return Values

No value is returned.

ImagickDraw::pathCurveToQuadraticBezierSmoothAbsolute

ImagickDraw::pathCurveToQuadraticBezierSmoothAbsolute -- Draws a quadratic Bezier curve

Description

bool **ImagickDraw::pathCurveToQuadraticBezierSmoothAbsolute** (float \$x, float \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a quadratic Bezier curve (using relative coordinates) from the current point to (x,y). The control point is assumed to be the reflection of the control point on the previous command relative to the current point. (If there is no previous command or if the previous command was not a DrawPathCurveToQuadraticBezierAbsolute, DrawPathCurveToQuadraticBezierRelative, DrawPathCurveToQuadraticBezierSmoothAbsolute or DrawPathCurveToQuadraticBezierSmoothRelative, assume the control point is coincident with the current point.). At the end of the command, the new current point becomes the final (x,y) coordinate pair used in the polybezier.

Parameters

x
ending x coordinate

y
ending y coordinate

Return Values

No value is returned.

ImagickDraw::pathCurveToQuadraticBezierSmoothRelative

ImagickDraw::pathCurveToQuadraticBezierSmoothRelative -- Draws a quadratic Bezier curve

Description

bool **ImagickDraw::pathCurveToQuadraticBezierSmoothRelative** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a quadratic Bezier curve (using relative coordinates) from the current point to (x, y). The control point is assumed to be the reflection of the control point on the previous command relative to the current point. (If there is no previous command or if the previous command was not a DrawPathCurveToQuadraticBezierAbsolute, DrawPathCurveToQuadraticBezierRelative, DrawPathCurveToQuadraticBezierSmoothAbsolute or DrawPathCurveToQuadraticBezierSmoothRelative, assume the control point is coincident with the current point). At the end of the command, the new current point becomes the final (x, y) coordinate pair used in the polybezier.

Parameters

x
ending x coordinate

y
ending y coordinate

Return Values

No value is returned.

ImagickDraw::pathCurveToRelative

ImagickDraw::pathCurveToRelative -- Draws a cubic Bezier curve

Description

bool **ImagickDraw::pathCurveToRelative** (float \$x1, float \$y1, float \$x2, float \$y2, float \$x, float \$y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a cubic Bezier curve from the current point to (x,y) using (x1,y1) as the control point at the beginning of the curve and (x2,y2) as the control point at the end of the curve using relative coordinates. At the end of the command, the new current point becomes the final (x,y) coordinate pair used in the polybezier.

Parameters

x1
x coordinate of starting control point

y1
y coordinate of starting control point

x2
x coordinate of ending control point

y2
y coordinate of ending control point

x
ending x coordinate

y
ending y coordinate

Return Values

No value is returned.

ImagickDraw::pathCurveToSmoothAbsolute

ImagickDraw::pathCurveToSmoothAbsolute -- Draws a cubic Bezier curve

Description

bool **ImagickDraw::pathCurveToSmoothAbsolute** (float x_2 , float y_2 , float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a cubic Bezier curve from the current point to (x,y) using absolute coordinates. The first control point is assumed to be the reflection of the second control point on the previous command relative to the current point. (If there is no previous command or if the previous command was not an DrawPathCurveToAbsolute, DrawPathCurveToRelative, DrawPathCurveToSmoothAbsolute or DrawPathCurveToSmoothRelative, assume the first control point is coincident with the current point.) (x2,y2) is the second control point (i.e., the control point at the end of the curve). At the end of the command, the new current point becomes the final (x,y) coordinate pair used in the polybezier.

Parameters

x_2
x coordinate of the second control point

y_2
y coordinate of the second control point

x
x coordinate of the ending point

y
y coordinate of the ending point

Return Values

No value is returned.

ImagickDraw::pathCurveToSmoothRelative

ImagickDraw::pathCurveToSmoothRelative -- Draws a cubic Bezier curve

Description

bool **ImagickDraw::pathCurveToSmoothRelative** (float x_2 , float y_2 , float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a cubic Bezier curve from the current point to (x,y) using relative coordinates. The first control point is assumed to be the reflection of the second control point on the previous command relative to the current point. (If there is no previous command or if the previous command was not an DrawPathCurveToAbsolute, DrawPathCurveToRelative, DrawPathCurveToSmoothAbsolute or DrawPathCurveToSmoothRelative, assume the first control point is coincident with the current point.) (x2,y2) is the second control point (i.e., the control point at the end of the curve). At the end of the command, the new current point becomes the final (x,y) coordinate pair used in the polybezier.

Parameters

x_2
x coordinate of the second control point

y_2
y coordinate of the second control point

x
x coordinate of the ending point

y
y coordinate of the ending point

Return Values

No value is returned.

ImagickDraw::pathEllipticArcAbsolute

ImagickDraw::pathEllipticArcAbsolute -- Draws an elliptical arc

Description

```
bool ImagickDraw::pathEllipticArcAbsolute ( float $rx, float $ry, float $
x_axis_rotation, bool $large_arc_flag, bool $sweep_flag, float $x, float $y )
```

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws an elliptical arc from the current point to (x, y) using absolute coordinates. The size and orientation of the ellipse are defined by two radii (rx, ry) and an xAxisRotation, which indicates how the ellipse as a whole is rotated relative to the current coordinate system. The center (cx, cy) of the ellipse is calculated automatically to satisfy the constraints imposed by the other parameters. largeArcFlag and sweepFlag contribute to the automatic calculations and help determine how the arc is drawn. If largeArcFlag is **TRUE** then draw the larger of the available arcs. If sweepFlag is true, then draw the arc matching a clock-wise rotation.

Parameters

rx
x radius

ry
y radius

x_axis_rotation
x axis rotation

large_arc_flag
large arc flag

sweep_flag
sweep flag

x
x coordinate

y
y coordinate

Return Values

No value is returned.

ImagickDraw::pathEllipticArcRelative

ImagickDraw::pathEllipticArcRelative -- Draws an elliptical arc

Description

```
bool ImagickDraw::pathEllipticArcRelative ( float $rx, float $ry, float $x_axis_rotation
, bool $large_arc_flag, bool $sweep_flag, float $x, float $y )
```

Warning

This function is currently not documented; only its argument list is available.

Draws an elliptical arc from the current point to (x, y) using relative coordinates. The size and orientation of the ellipse are defined by two radii (rx, ry) and an xAxisRotation, which indicates how the ellipse as a whole is rotated relative to the current coordinate system. The center (cx, cy) of the ellipse is calculated automatically to satisfy the constraints imposed by the other parameters. largeArcFlag and sweepFlag contribute to the automatic calculations and help determine how the arc is drawn. If largeArcFlag is **TRUE** then draw the larger of the available arcs. If sweepFlag is true, then draw the arc matching a clock-wise rotation.

Parameters

rx
x radius

ry
y radius

x_axis_rotation
x axis rotation

large_arc_flag
large arc flag

sweep_flag
sweep flag

x
x coordinate

y
y coordinate

Return Values

No value is returned.

ImagickDraw::pathFinish

ImagickDraw::pathFinish -- Terminates the current path

Description

bool **ImagickDraw::pathFinish** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Terminates the current path.

Return Values

No value is returned.

ImagickDraw::pathLineToAbsolute

ImagickDraw::pathLineToAbsolute -- Draws a line path

Description

bool **ImagickDraw::pathLineToAbsolute** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a line path from the current point to the given coordinate using absolute coordinates. The coordinate then becomes the new current point.

Parameters

x
starting x coordinate

y
ending x coordinate

Return Values

No value is returned.

ImagickDraw::pathLineToHorizontalAbsolute

ImagickDraw::pathLineToHorizontalAbsolute -- Draws a horizontal line path

Description

bool **ImagickDraw::pathLineToHorizontalAbsolute** (float \$x)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a horizontal line path from the current point to the target point using absolute coordinates. The target point then becomes the new current point.

Parameters

x
x coordinate

Return Values

No value is returned.

ImagickDraw::pathLineToHorizontalRelative

ImagickDraw::pathLineToHorizontalRelative -- Draws a horizontal line

Description

bool **ImagickDraw::pathLineToHorizontalRelative** (float \$x)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a horizontal line path from the current point to the target point using relative coordinates. The target point then becomes the new current point.

Parameters

x
x coordinate

Return Values

No value is returned.

ImagickDraw::pathLineToRelative

ImagickDraw::pathLineToRelative -- Draws a line path

Description

bool **ImagickDraw::pathLineToRelative** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a line path from the current point to the given coordinate using relative coordinates. The coordinate then becomes the new current point.

Parameters

x
starting x coordinate

y
starting y coordinate

Return Values

No value is returned.

ImagickDraw::pathLineToVerticalAbsolute

ImagickDraw::pathLineToVerticalAbsolute -- Draws a vertical line

Description

bool **ImagickDraw::pathLineToVerticalAbsolute** (float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a vertical line path from the current point to the target point using absolute coordinates. The target point then becomes the new current point.

Parameters

y
y coordinate

Return Values

No value is returned.

ImagickDraw::pathLineToVerticalRelative

ImagickDraw::pathLineToVerticalRelative -- Draws a vertical line path

Description

bool **ImagickDraw::pathLineToVerticalRelative** (float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a vertical line path from the current point to the target point using relative coordinates. The target point then becomes the new current point.

Parameters

y
y coordinate

Return Values

No value is returned.

ImagickDraw::pathMoveToAbsolute

ImagickDraw::pathMoveToAbsolute -- Starts a new sub-path

Description

bool **ImagickDraw::pathMoveToAbsolute** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Starts a new sub-path at the given coordinate using absolute coordinates. The current point then becomes the specified coordinate.

Parameters

x
x coordinate of the starting point

y
y coordinate of the starting point

Return Values

No value is returned.

ImagickDraw::pathMoveToRelative

ImagickDraw::pathMoveToRelative -- Starts a new sub-path

Description

bool **ImagickDraw::pathMoveToRelative** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Starts a new sub-path at the given coordinate using relative coordinates. The current point then becomes the specified coordinate.

Parameters

x
target x coordinate

y
target y coordinate

Return Values

No value is returned.

ImagickDraw::pathStart

ImagickDraw::pathStart -- Declares the start of a path drawing list

Description

bool **ImagickDraw::pathStart** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Declares the start of a path drawing list which is terminated by a matching DrawPathFinish() command. All other DrawPath commands must be enclosed between a DrawPathStart() and a DrawPathFinish() command. This is because path drawing commands are subordinate commands and they do not function by themselves.

Return Values

No value is returned.

ImagickDraw::point

ImagickDraw::point -- Draws a point

Description

bool **ImagickDraw::point** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a point using the current stroke color and stroke thickness at the specified coordinates.

Parameters

x
point's x coordinate

y
point's y coordinate

Return Values

No value is returned.

ImagickDraw::polygon

ImagickDraw::polygon -- Draws a polygon

Description

bool **ImagickDraw::polygon** (array \$coordinates)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a polygon using the current stroke, stroke width, and fill color or texture, using the specified array of coordinates.

Parameters

coordinates

multidimensional array like array(array('x' => 3, 'y' => 4), array('x' => 2, 'y' => 6));

Return Values

Returns **TRUE** on success.

ImagickDraw::polyline

ImagickDraw::polyline -- Draws a polyline

Description

bool **ImagickDraw::polyline** (array \$coordinates)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a polyline using the current stroke, stroke width, and fill color or texture, using the specified array of coordinates.

Parameters

coordinates

array of x and y coordinates: array(array('x' => 4, 'y' => 6), array('x' => 8, 'y' => 10))

Return Values

Returns **TRUE** on success.

ImagickDraw::pop

ImagickDraw::pop -- Destroys the current ImagickDraw in the stack, and returns to the previously pushed ImagickDraw

Description

bool **ImagickDraw::pop** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Destroys the current ImagickDraw in the stack, and returns to the previously pushed ImagickDraw. Multiple ImagickDraws may exist. It is an error to attempt to pop more ImagickDraws than have been pushed, and it is proper form to pop all ImagickDraws which have been pushed.

Return Values

Returns **TRUE** on success and false on failure.

ImagickDraw::popClipPath

ImagickDraw::popClipPath -- Terminates a clip path definition

Description

bool **ImagickDraw::popClipPath** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Terminates a clip path definition.

Return Values

No value is returned.

ImagickDraw::popDefs

ImagickDraw::popDefs -- Terminates a definition list

Description

bool **ImagickDraw::popDefs** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Terminates a definition list.

Return Values

No value is returned.

ImagickDraw::popPattern

ImagickDraw::popPattern -- Terminates a pattern definition

Description

bool **ImagickDraw::popPattern** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Terminates a pattern definition.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

ImagickDraw::push

ImagickDraw::push -- Clones the current ImagickDraw and pushes it to the stack

Description

bool **ImagickDraw::push** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Clones the current ImagickDraw to create a new ImagickDraw, which is then added to the ImagickDraw stack. The original drawing ImagickDraw(s) may be returned to by invoking pop(). The ImagickDraws are stored on a ImagickDraw stack. For every Pop there must have already been an equivalent Push.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

ImagickDraw::pushClipPath

ImagickDraw::pushClipPath -- Starts a clip path definition

Description

bool **ImagickDraw::pushClipPath** (string *\$clip_mask_id*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Starts a clip path definition which is comprized of any number of drawing commands and terminated by a ImagickDraw::popClipPath() command.

Parameters

clip_mask_id
Clip mask Id

Return Values

No value is returned.

ImagickDraw::pushDefs

ImagickDraw::pushDefs -- Indicates that following commands create named elements for early processing

Description

bool **ImagickDraw::pushDefs** (void)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Indicates that commands up to a terminating ImagickDraw::popDefs() command create named elements (e.g. clip-paths, textures, etc.) which may safely be processed earlier for the sake of efficiency.

Return Values

No value is returned.

ImagickDraw::pushPattern

ImagickDraw::pushPattern -- Indicates that subsequent commands up to a ImagickDraw::opPattern() command comprise the definition of a named pattern

Description

bool **ImagickDraw::pushPattern** (string \$pattern_id, float \$x, float \$y, float \$width, float \$height)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Indicates that subsequent commands up to a DrawPopPattern() command comprise the definition of a named pattern. The pattern space is assigned top left corner coordinates, a width and height, and becomes its own drawing space. Anything which can be drawn may be used in a pattern definition. Named patterns may be used as stroke or brush definitions.

Parameters

pattern_id
the pattern Id

x
x coordinate of the top-left corner

y
y coordinate of the top-left corner

width
width of the pattern

height
height of the pattern

Return Values

Returns **TRUE** on success or **FALSE** on failure.

ImagickDraw::rectangle

ImagickDraw::rectangle -- Draws a rectangle

Description

bool **ImagickDraw::rectangle** (float *\$x1*, float *\$y1*, float *\$x2*, float *\$y2*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a rectangle given two coordinates and using the current stroke, stroke width, and fill settings.

Parameters

x1
x coordinate of the top left corner

y1
y coordinate of the top left corner

x2
x coordinate of the bottom right corner

y2
y coordinate of the bottom right corner

Return Values

No value is returned.

ImagickDraw::render

ImagickDraw::render -- Renders all preceding drawing commands onto the image

Description

bool **ImagickDraw::render** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Renders all preceding drawing commands onto the image.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

ImagickDraw::rotate

ImagickDraw::rotate -- Applies the specified rotation to the current coordinate space

Description

bool **ImagickDraw::rotate** (float \$degrees)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Applies the specified rotation to the current coordinate space.

Parameters

degrees
degrees to rotate

Return Values

No value is returned.

ImagickDraw::roundRectangle

ImagickDraw::roundRectangle -- Draws a rounded rectangle

Description

bool **ImagickDraw::roundRectangle** (float \$x1, float \$y1, float \$x2, float \$y2, float \$rx, float \$ry)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Draws a rounded rectangle given two coordinates, x & y corner radiuses and using the current stroke, stroke width, and fill settings.

Parameters

x1
x coordinate of the top left corner

y1
y coordinate of the top left corner

x2
x coordinate of the bottom right

y2
y coordinate of the bottom right

rx
x rounding

ry
y rounding

Return Values

No value is returned.

ImagickDraw::scale

ImagickDraw::scale -- Adjusts the scaling factor

Description

bool **ImagickDraw::scale** (float x , float y)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Adjusts the scaling factor to apply in the horizontal and vertical directions to the current coordinate space.

Parameters

x
horizontal factor

y
vertical factor

Return Values

No value is returned.

ImagickDraw::setClipPath

ImagickDraw::setClipPath -- Associates a named clipping path with the image

Description

bool **ImagickDraw::setClipPath** (string *\$clip_mask*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Associates a named clipping path with the image. Only the areas drawn on by the clipping path will be modified as long as it remains in effect.

Parameters

clip_mask
the clipping path name

Return Values

No value is returned.

ImagickDraw::setClipRule

ImagickDraw::setClipRule -- Set the polygon fill rule to be used by the clipping path

Description

bool **ImagickDraw::setClipRule** (int *\$fill_rule*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Set the polygon fill rule to be used by the clipping path.

Parameters

fill_rule
FILLRULE_ constant

Return Values

No value is returned.

ImagickDraw::setClipUnits

ImagickDraw::setClipUnits -- Sets the interpretation of clip path units

Description

bool **ImagickDraw::setClipUnits** (int *\$clip_units*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the interpretation of clip path units.

Parameters

clip_units
the number of clip units

Return Values

No value is returned.

ImagickDraw::setFillAlpha

ImagickDraw::setFillAlpha -- Sets the opacity to use when drawing using the fill color or fill texture

Description

bool **ImagickDraw::setFillAlpha** (float *\$opacity*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the opacity to use when drawing using the fill color or fill texture. Fully opaque is 1.0.

Parameters

opacity
fill alpha

Return Values

No value is returned.

ImagickDraw::setFillColor

ImagickDraw::setFillColor -- Sets the fill color to be used for drawing filled objects

Description

bool **ImagickDraw::setFillColor** ([ImagickPixel](#) \$fill_pixel)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the fill color to be used for drawing filled objects.

Parameters

fill_pixel
ImagickPixel to use to set the color

Return Values

No value is returned.

ImagickDraw::setFillOpacity

ImagickDraw::setFillOpacity -- Sets the opacity to use when drawing using the fill color or fill texture

Description

bool **ImagickDraw::setFillOpacity** (float *\$fillOpacity*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the opacity to use when drawing using the fill color or fill texture. Fully opaque is 1.0.

Parameters

fillOpacity
the fill opacity

Return Values

No value is returned.

ImagickDraw::setFillPatternURL

ImagickDraw::setFillPatternURL -- Sets the URL to use as a fill pattern for filling objects

Description

bool **ImagickDraw::setFillPatternURL** (string *\$fill_url*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the URL to use as a fill pattern for filling objects. Only local URLs ("#identifier") are supported at this time. These local URLs are normally created by defining a named fill pattern with DrawPushPattern/DrawPopPattern.

Parameters

fill_url

URL to use to obtain fill pattern.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

ImagickDraw::setFillRule

ImagickDraw::setFillRule -- Sets the fill rule to use while drawing polygons

Description

bool **ImagickDraw::setFillRule** (int *\$fill_rule*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the fill rule to use while drawing polygons.

Parameters

fill_rule
FILLRULE_ constant

Return Values

No value is returned.

ImagickDraw::setFont

ImagickDraw::setFont -- Sets the fully-specified font to use when annotating with text

Description

bool **ImagickDraw::setFont** (string *\$font_name*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the fully-specified font to use when annotating with text.

Parameters

font_name

Return Values

Returns **TRUE** on success.

ImagickDraw::setFontFamily

ImagickDraw::setFontFamily -- Sets the font family to use when annotating with text

Description

bool **ImagickDraw::setFontFamily** (string \$font_family)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the font family to use when annotating with text.

Parameters

font_family
the font family

Return Values

Returns **TRUE** on success.

ImagickDraw::setFontSize

ImagickDraw::setFontSize -- Sets the font pointsize to use when annotating with text

Description

bool **ImagickDraw::setFontSize** (float *\$pointsize*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the font pointsize to use when annotating with text.

Parameters

pointsize
the point size

Return Values

No value is returned.

ImagickDraw::setFontStretch

ImagickDraw::setFontStretch -- Sets the font stretch to use when annotating with text

Description

bool **ImagickDraw::setFontStretch** (int \$fontStretch)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the font stretch to use when annotating with text. The AnyStretch enumeration acts as a wild-card "don't care" option.

Parameters

fontStretch
STRETCH_ constant

Return Values

No value is returned.

ImagickDraw::setFontStyle

ImagickDraw::setFontStyle -- Sets the font style to use when annotating with text

Description

bool **ImagickDraw::setFontStyle** (int *\$style*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the font style to use when annotating with text. The AnyStyle enumeration acts as a wild-card "don't care" option.

Parameters

style
STYLETYPE_ constant

Return Values

No value is returned.

ImagickDraw::setFontWeight

ImagickDraw::setFontWeight -- Sets the font weight

Description

bool **ImagickDraw::setFontWeight** (int *\$font_weight*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the font weight to use when annotating with text.

Parameters

font_weight

Return Values

ImagickDraw::setGravity

ImagickDraw::setGravity -- Sets the text placement gravity

Description

bool **ImagickDraw::setGravity** (int *\$gravity*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the text placement gravity to use when annotating with text.

Parameters

gravity
GRAVITY_ constant

Return Values

No value is returned.

ImagickDraw::setStrokeAlpha

ImagickDraw::setStrokeAlpha -- Specifies the opacity of stroked object outlines

Description

bool **ImagickDraw::setStrokeAlpha** (float \$opacity)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Specifies the opacity of stroked object outlines.

Parameters

opacity
opacity

Return Values

No value is returned.

ImagickDraw::setStrokeAntialias

ImagickDraw::setStrokeAntialias -- Controls whether stroked outlines are antialiased

Description

bool **ImagickDraw::setStrokeAntialias** (bool *\$stroke_antialias*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Controls whether stroked outlines are antialiased. Stroked outlines are antialiased by default. When antialiasing is disabled stroked pixels are thresholded to determine if the stroke color or underlying canvas color should be used.

Parameters

stroke_antialias
the antialias setting

Return Values

No value is returned.

ImagickDraw::setStrokeColor

ImagickDraw::setStrokeColor -- Sets the color used for stroking object outlines

Description

bool **ImagickDraw::setStrokeColor** ([ImagickPixel](#) \$stroke_pixel)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the color used for stroking object outlines.

Parameters

stroke_pixel
the stroke color

Return Values

No value is returned.

ImagickDraw::setStrokeDashArray

ImagickDraw::setStrokeDashArray -- Specifies the pattern of dashes and gaps used to stroke paths

Description

bool **ImagickDraw::setStrokeDashArray** (array \$dashArray)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Specifies the pattern of dashes and gaps used to stroke paths. The strokeDashArray represents an array of numbers that specify the lengths of alternating dashes and gaps in pixels. If an odd number of values is provided, then the list of values is repeated to yield an even number of values. To remove an existing dash array, pass a zero number_elements argument and null dash_array. A typical strokeDashArray_ array might contain the members 5 3 2.

Parameters

dashArray
array of floats

Return Values

Returns **TRUE** on success.

ImagickDraw::setStrokeDashOffset

ImagickDraw::setStrokeDashOffset -- Specifies the offset into the dash pattern to start the dash

Description

bool **ImagickDraw::setStrokeDashOffset** (float *\$dash_offset*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Specifies the offset into the dash pattern to start the dash.

Parameters

dash_offset
dash offset

Return Values

No value is returned.

ImagickDraw::setStrokeLineCap

ImagickDraw::setStrokeLineCap -- Specifies the shape to be used at the end of open subpaths when they are stroked

Description

bool **ImagickDraw::setStrokeLineCap** (int *\$linecap*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Specifies the shape to be used at the end of open subpaths when they are stroked.

Parameters

linecap
LINECAP_ constant

Return Values

No value is returned.

ImagickDraw::setStrokeLineJoin

ImagickDraw::setStrokeLineJoin -- Specifies the shape to be used at the corners of paths when they are stroked

Description

bool **ImagickDraw::setStrokeLineJoin** (int *\$linejoin*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Specifies the shape to be used at the corners of paths (or other vector shapes) when they are stroked.

Parameters

linejoin
LINEJOIN_ constant

Return Values

No value is returned.

ImagickDraw::setStrokeMiterLimit

ImagickDraw::setStrokeMiterLimit -- Specifies the miter limit

Description

bool **ImagickDraw::setStrokeMiterLimit** (int *\$miterlimit*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Specifies the miter limit. When two line segments meet at a sharp angle and miter joins have been specified for 'lineJoin', it is possible for the miter to extend far beyond the thickness of the line stroking the path. The miterLimit' imposes a limit on the ratio of the miter length to the 'lineWidth'.

Parameters

miterlimit
the miter limit

Return Values

No value is returned.

ImagickDraw::setStrokeOpacity

ImagickDraw::setStrokeOpacity -- Specifies the opacity of stroked object outlines

Description

bool **ImagickDraw::setStrokeOpacity** (float *\$stroke_opacity*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Specifies the opacity of stroked object outlines.

Parameters

stroke_opacity
stroke opacity. 1.0 is fully opaque

Return Values

No value is returned.

ImagickDraw::setStrokePatternURL

ImagickDraw::setStrokePatternURL -- Sets the pattern used for stroking object outlines

Description

bool **ImagickDraw::setStrokePatternURL** (string *\$stroke_url*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the pattern used for stroking object outlines.

Parameters

stroke_url
stroke URL

Return Values

imagick.imagickdraw.return.success;

ImagickDraw::setStrokeWidth

ImagickDraw::setStrokeWidth -- Sets the width of the stroke used to draw object outlines

Description

bool **ImagickDraw::setStrokeWidth** (float *\$stroke_width*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the width of the stroke used to draw object outlines.

Parameters

stroke_width
stroke width

Return Values

No value is returned.

ImagickDraw::setTextAlignment

ImagickDraw::setTextAlignment -- Specifies a text alignment

Description

bool **ImagickDraw::setTextAlignment** (int *\$alignment*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Specifies a text alignment to be applied when annotating with text.

Parameters

alignment
ALIGN_ constant

Return Values

No value is returned.

ImagickDraw::setTextAntialias

ImagickDraw::setTextAntialias -- Controls whether text is antialiased

Description

bool **ImagickDraw::setTextAntialias** (bool *\$antiAlias*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Controls whether text is antialiased. Text is antialiased by default.

Parameters

antiAlias

Return Values

No value is returned.

ImagickDraw::setTextDecoration

ImagickDraw::setTextDecoration -- Specifies a decoration

Description

bool **ImagickDraw::setTextDecoration** (int *\$decoration*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Specifies a decoration to be applied when annotating with text.

Parameters

decoration
DECORATION_ constant

Return Values

No value is returned.

ImagickDraw::setTextEncoding

ImagickDraw::setTextEncoding -- Specifies specifies the text code set

Description

bool **ImagickDraw::setTextEncoding** (string *\$encoding*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Specifies specifies the code set to use for text annotations. The only character encoding which may be specified at this time is "UTF-8" for representing Unicode as a sequence of bytes. Specify an empty string to set text encoding to the system's default. Successful text annotation using Unicode may require fonts designed to support Unicode.

Parameters

encoding

the encoding name

Return Values

No value is returned.

ImagickDraw::setTextUnderColor

ImagickDraw::setTextUnderColor -- Specifies the color of a background rectangle

Description

bool **ImagickDraw::setTextUnderColor** ([ImagickPixel](#) \$under_color)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Specifies the color of a background rectangle to place under text annotations.

Parameters

under_color
the under color

Return Values

No value is returned.

ImagickDraw::setVectorGraphics

ImagickDraw::setVectorGraphics -- Sets the vector graphics

Description

bool **ImagickDraw::setVectorGraphics** (string *\$xml*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the vector graphics associated with the specified ImagickDraw object. Use this method with ImagickDraw::getVectorGraphics() as a method to persist the vector graphics state.

Parameters

xml

xml containing the vector graphics

Return Values

Returns **TRUE** on success or **FALSE** on failure.

ImagickDraw::setViewbox

ImagickDraw::setViewbox -- Sets the overall canvas size

Description

bool **ImagickDraw::setViewbox** (int *\$x1*, int *\$y1*, int *\$x2*, int *\$y2*)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the overall canvas size to be recorded with the drawing vector data. Usually this will be specified using the same size as the canvas image. When the vector data is saved to SVG or MVG formats, the viewbox is use to specify the size of the canvas image that a viewer will render the vector data on.

Parameters

x1
left x coordinate

y1
left y coordinate

x2
right x coordinate

y2
right y coordinate

Return Values

No value is returned.

ImagickDraw::skewX

ImagickDraw::skewX -- Skews the current coordinate system in the horizontal direction

Description

bool **ImagickDraw::skewX** (float *\$degrees*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Skews the current coordinate system in the horizontal direction.

Parameters

degrees
degrees to skew

Return Values

No value is returned.

ImagickDraw::skewY

ImagickDraw::skewY -- Skews the current coordinate system in the vertical direction

Description

bool **ImagickDraw::skewY** (float *\$degrees*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Skews the current coordinate system in the vertical direction.

Parameters

degrees
degrees to skew

Return Values

No value is returned.

ImagickDraw::translate

ImagickDraw::translate -- Applies a translation to the current coordinate system

Description

bool **ImagickDraw::translate** (float x , float y)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Applies a translation to the current coordinate system which moves the coordinate system origin to the specified coordinate.

Parameters

x
horizontal translation

y
vertical translation

Return Values

No value is returned.

The MagickPixel class

Class synopsis

| |
|--------------------|
| MagickPixel |
|--------------------|

```
MagickPixel {  
    bool MagickPixel::clear ( void )  
  
    MagickPixel MagickPixel::__construct ( [ string $color ] )  
  
    bool MagickPixel::destroy ( void )  
  
    array MagickPixel::getColor ( [ bool $normalized ] )  
  
    string MagickPixel::getColorAsString ( void )  
  
    int MagickPixel::getColorCount ( void )  
  
    float MagickPixel::getColorValue ( int $color )  
  
    array MagickPixel::getHSL ( void )  
  
    bool MagickPixel::isSimilar ( ImagickPixel $color, float $fuzz )  
  
    bool MagickPixel::setColor ( string $color )  
  
    bool MagickPixel::setColorValue ( int $color, float $value )  
  
    bool MagickPixel::setHSL ( float $hue, float $saturation, float $luminosity )  
}
```

ImagickPixel::clear

ImagickPixel::clear -- Clears resources associated with this object

Description

bool **ImagickPixel::clear** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Clears the ImagickPixel object, leaving it in a fresh state. This also unsets any color associated with the object.

Return Values

Returns **TRUE** on success.

ImagickPixel::__construct

ImagickPixel::__construct -- The ImagickPixel constructor

Description

[ImagickPixel](#) **ImagickPixel::__construct** ([string *\$color*])

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Constructs an ImagickPixel object. If a color is specified, the object is constructed and then initialised with that color before being returned.

Parameters

color

The optional color string to use as the initial value of this object.

Return Values

Returns an ImagickPixel object on success, throwing ImagickPixelException on failure.

ImagickPixel::destroy

ImagickPixel::destroy -- Deallocates resources associated with this object

Description

bool **ImagickPixel::destroy** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Deallocates any resources used by the ImagickPixel object, and unsets any associated color. The object should not be used after the destroy function has been called.

Return Values

Returns **TRUE** on success.

ImagickPixel::getColor

ImagickPixel::getColor -- Returns the color

Description

array **ImagickPixel::getColor** ([bool *\$normalized*])

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Returns the color described by the ImagickPixel object, as an array. If the color has an opacity channel set, this is provided as a fourth value in the list.

Parameters

normalized

Normalize the color values

Return Values

An array of channel values, each normalized if **TRUE** is given as param. Throws ImagickPixelException on error.

ImagickPixel::getColorAsString

ImagickPixel::getColorAsString -- Returns the color as a string

Description

string **ImagickPixel::getColorAsString** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the color of the ImagickPixel object as a string.

Parameters

Return Values

Returns the color of the ImagickPixel object as a string.

ImagickPixel::getColorCount

ImagickPixel::getColorCount -- Returns the color count associated with this color

Description

int **ImagickPixel::getColorCount** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the color count associated with this color.

Return Values

Returns the color count as an integer on success, throws ImagickPixelException on failure.

ImagickPixel::getColorValue

ImagickPixel::getColorValue -- Gets the normalized value of the provided color channel

Description

float **ImagickPixel::getColorValue** (int *\$color*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Retrieves the value of the color channel specified, as a floating-point number between 0 and 1.

Parameters

color

The channel to check, specified as one of the Imagick channel constants.

Return Values

The value of the channel, as a normalized floating-point number, throwing ImagickPixelException on error.

ImagickPixel::getHSL

ImagickPixel::getHSL -- Returns the normalized HSL color of the ImagickPixel object

Description

array **ImagickPixel::getHSL** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the normalized HSL color described by the ImagickPixel object, with each of the three values as floating point numbers between 0.0 and 1.0.

Return Values

Returns the HSL value in an array with the keys "hue", "saturation", and "luminosity".
Throws ImagickPixelException on failure.

ImagickPixel::isSimilar

ImagickPixel::isSimilar -- Check the distance between this color and another

Description

bool **ImagickPixel::isSimilar** ([ImagickPixel](#) \$color, float \$fuzz)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Checks the distance between the color described by this ImagickPixel object and that of the provided object, by plotting their RGB values on the color cube. If the distance between the two points is less than the fuzz value given, the colors are similar.

Parameters

color

The ImagickPixel object to compare this object against.

fuzz

The maximum distance within which to consider these colors as similar. The theoretical maximum for this value is the square root of three (1.732).

Return Values

Returns **TRUE** on success.

ImagickPixel::setColor

ImagickPixel::setColor -- Sets the color

Description

bool **ImagickPixel::setColor** (string \$color)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the color described by the ImagickPixel object, with a string (e.g. "blue", "#0000ff", "rgb(0,0,255)", "cmyk(100,100,100,10)", etc.).

Parameters

color

The color definition to use in order to initialise the ImagickPixel object.

Return Values

Returns **TRUE** if the specified color was set, **FALSE** otherwise.

ImagickPixel::setColorValue

ImagickPixel::setColorValue -- Sets the normalized value of one of the channels

Description

bool **ImagickPixel::setColorValue** (int \$color, float \$value)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the value of the specified channel of this object to the provided value, which should be between 0 and 1. This function can be used to provide an opacity channel to an ImagickPixel object.

Parameters

color

One of the Imagick channel color constants.

value

The value to set this channel to, ranging from 0 to 1.

Return Values

Returns **TRUE** on success.

ImagickPixel::setHSL

ImagickPixel::setHSL -- Sets the normalized HSL color

Description

bool **ImagickPixel::setHSL** (float \$hue, float \$saturation, float \$luminosity)

| Warning |
|---|
| This function is currently not documented; only its argument list is available. |

Sets the color described by the ImagickPixel object using normalized values for hue, saturation and luminosity.

Parameters

hue

The normalized value for hue, described as a fractional arc (between 0 and 1) of the hue circle, where the zero value is red.

saturation

The normalized value for saturation, with 1 as full saturation.

luminosity

The normalized value for luminosity, on a scale from black at 0 to white at 1, with the full HS value at 0.5 luminosity.

Return Values

Returns **TRUE** on success.

The ImagickPixelIterator class

Class synopsis

| |
|-----------------------------|
| ImagickPixelIterator |
|-----------------------------|

```
ImagickPixelIterator {  
    bool ImagickPixelIterator::clear ( void )  
  
    ImagickPixelIterator ImagickPixelIterator::__construct ( Imagick $wand )  
  
    bool ImagickPixelIterator::destroy ( void )  
  
    array ImagickPixelIterator::getCurrentIteratorRow ( void )  
  
    int ImagickPixelIterator::getIteratorRow ( void )  
  
    array ImagickPixelIterator::getNextIteratorRow ( void )  
  
    array ImagickPixelIterator::getPreviousIteratorRow ( void )  
  
    bool ImagickPixelIterator::newPixelIterator ( Imagick $wand )  
  
    bool ImagickPixelIterator::newPixelRegionIterator ( Imagick $wand, int $x, int $y,  
    int $columns, int $rows )  
  
    bool ImagickPixelIterator::resetIterator ( void )  
  
    bool ImagickPixelIterator::setIteratorFirstRow ( void )  
  
    bool ImagickPixelIterator::setIteratorLastRow ( void )  
  
    bool ImagickPixelIterator::setIteratorRow ( int $row )  
  
    bool ImagickPixelIterator::syncIterator ( void )  
}
```

ImagickPixelIterator::clear

ImagickPixelIterator::clear -- Clear resources associated with a PixelIterator

Description

bool **ImagickPixelIterator::clear** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Clear resources associated with a PixelIterator.

Return Values

Returns **TRUE** on success.

ImagickPixelIterator::__construct

ImagickPixelIterator::__construct -- The ImagickPixelIterator constructor

Description

[ImagickPixelIterator](#) **ImagickPixelIterator::__construct** ([Imagick](#) \$wand)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

The ImagickPixelIterator constructor

Return Values

Returns **TRUE** on success.

ImagickPixelIterator::destroy

ImagickPixelIterator::destroy -- Deallocates resources associated with a PixelIterator

Description

bool **ImagickPixelIterator::destroy** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Deallocates resources associated with a PixelIterator.

Return Values

Returns **TRUE** on success.

ImagickPixelIterator::getCurrentIteratorRow

ImagickPixelIterator::getCurrentIteratorRow -- Returns the current row of ImagickPixel objects

Description

array **ImagickPixelIterator::getCurrentIteratorRow** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the current row as an array of ImagickPixel objects from the pixel iterator.

Return Values

Returns a row as an array of ImagickPixel objects that can themselves be iterated.

ImagickPixelIterator::getIteratorRow

ImagickPixelIterator::getIteratorRow -- Returns the current pixel iterator row

Description

int **ImagickPixelIterator::getIteratorRow** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the current pixel iterator row.

Return Values

Returns the integer offset of the row, throwing ImagickPixelIteratorException on error.

ImagickPixelIterator::getNextIteratorRow

ImagickPixelIterator::getNextIteratorRow -- Returns the next row of the pixel iterator

Description

array **ImagickPixelIterator::getNextIteratorRow** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the next row as an array of pixel wands from the pixel iterator.

Return Values

Returns a row as an array of ImagickPixel objects, throwing ImagickPixelIteratorException on error.

ImagickPixelIterator::getPreviousIteratorRow

ImagickPixelIterator::getPreviousIteratorRow -- Returns the previous row

Description

array **ImagickPixelIterator::getPreviousIteratorRow** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns the previous row as an array of pixel wands from the pixel iterator.

Return Values

Returns the previous row as an array of ImagickPixelWand objects from the ImagickPixelIterator, throwing ImagickPixelIteratorException on error.

ImagickPixelIterator::newPixelIterator

ImagickPixelIterator::newPixelIterator -- Returns a new pixel iterator

Description

bool **ImagickPixelIterator::newPixelIterator** ([Imagick](#) \$wand)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns a new pixel iterator.

Return Values

Returns **TRUE** on success. Throwing ImagickPixelIteratorException.

ImagickPixelIterator::newPixelRegionIterator

ImagickPixelIterator::newPixelRegionIterator -- Returns a new pixel iterator

Description

bool **ImagickPixelIterator::newPixelRegionIterator** ([Imagick](#) \$wand, int \$x, int \$y, int \$columns, int \$rows)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Returns a new pixel iterator.

Parameters

wand

x

y

columns

rows

Return Values

Returns a new ImagickPixelIterator on success; on failure, throws ImagickPixelIteratorException.

ImagickPixelIterator::resetIterator

ImagickPixelIterator::resetIterator -- Resets the pixel iterator

Description

bool **ImagickPixelIterator::resetIterator** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Resets the pixel iterator. Use it in conjunction with ImagickPixelIterator::getNextIteratorRow() to iterate over all the pixels in a pixel container.

Return Values

Returns **TRUE** on success.

ImagickPixelIterator::setIteratorFirstRow

ImagickPixelIterator::setIteratorFirstRow -- Sets the pixel iterator to the first pixel row

Description

bool **ImagickPixelIterator::setIteratorFirstRow** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the pixel iterator to the first pixel row.

Return Values

Returns **TRUE** on success.

ImagickPixelIterator::setIteratorLastRow

ImagickPixelIterator::setIteratorLastRow -- Sets the pixel iterator to the last pixel row

Description

bool **ImagickPixelIterator::setIteratorLastRow** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Sets the pixel iterator to the last pixel row.

Return Values

Returns **TRUE** on success.

ImagickPixelIterator::setIteratorRow

ImagickPixelIterator::setIteratorRow -- Set the pixel iterator row

Description

bool **ImagickPixelIterator::setIteratorRow** (int *\$row*)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Set the pixel iterator row.

Parameters

row

Return Values

Returns **TRUE** on success.

ImagickPixelIterator::syncIterator

ImagickPixelIterator::syncIterator -- Syncs the pixel iterator

Description

bool **ImagickPixelIterator::syncIterator** (void)

| |
|---|
| Warning |
| This function is currently not documented; only its argument list is available. |

Syncs the pixel iterator.

Return Values

Returns **TRUE** on success.