ClickBlocks\Utils\Picture

General information

Inheritance	no
Subclasses	no
Interfaces	no
Source	Framework/utils/picture.php

Easy to use class that enables to crop, scale and rotate any image. You can also use this class to get general information about an image.

Public static methods

rgb2int()

```
public static integer rgb2int(integer $red, integer $green, integer $blue, float $alpha = 0)
```

$red	integer	value of red component.
$green	integer	value of green component.
$blue	integer	value of blue component.
$alpha	integer	a value between 0 and 1. 0 indicates completely opaque while 1 indicates completely transparent.

Returns a color identifier representing the color composed of the given RGB components and the transparency parameter alpha. The colors parameters are integers between 0 and 255 or hexadecimals between 0x00 and 0xFF.

Public non-static methods

__construct()

```
public void __construct(string $image)
```

$image	string	full path to the image file.

Constructor. Reads information about the specified image.

isRGB()

```
public boolean isRGB()
```

Returns TRUE if the given image uses the RGB color model and FALSE otherwise.

isCMYK()

```
public boolean isCMYK()
```

Returns TRUE if the given image uses the CMYK color model and FALSE otherwise.

getWidth()

```
public integer getWidth()
```

Returns the image width.

getHeight()

```
public integer getHeight()
```

Returns the image height.

getExtension()

```
public string getExtension(boolean $includeDot = false)
```

$includeDot	boolean	determines whether to prepend a dot to the extension or not.

Returns suitable extension of the image file. The suitable extension is determined by the mime type of the image.

getSize()

```
public integer getSize()
```

Returns size (in bytes) of the image.

getMimeType()

```
public string getMimeType()
```

Returns mime type of the image.

getResource()

```
public resource getResource()
```

Returns an image resource identifier.

getColorDepth()

```
public integer getColorDepth()
```

Returns number of bits for each color of the image.

rotate()

```
public resource|boolean rotate(float $angle, integer $bgcolor = 0, integer $interpolationMethod = IMG_BILINEAR_FIXED)
```

$angle	float	rotation angle, in degrees. The rotation angle is interpreted as the number of degrees to rotate the image anticlockwise.
$bgcolor	integer	specifies the color of the uncovered zone after the rotation. It consists of RGB components and can also have alpha component for true color images.
$interpolationMethod	integer	the interpolation method. See more details here.

Rotates the image with a given angle. The method returns an image resource for the rotated image, or FALSE on failure. Example:

```
// Reads image info.
$pic = new Picture('example.png');
// Rotates the image.
$img = $pic->rotate(45, Picture::rgb2int(0xFF, 0, 0xFF, 0.5), IMG_BICUBIC);
// Sends the rotated image to browser.
header('Content-Type: image/png');
imagepng($img);
```

> Note that after the original image is rotated, it is not stored in the file. The changed image exists only in memory.

crop()

```
public resource|boolean crop(integer $left, integer $top, integer $width, integer $height, integer $bgcolor = 0, boolean $fixedSiz
```

$left	integer	the X coordinate of the image fragment. It can be negative number.
$top	integer	the Y coordinate of the image fragment. It can be negative number.
$width	integer	the width of the image fragment.
$height	integer	the width of the image fragment.
$bgcolor	integer	the background color of the uncovered zone of the cropped image.
$fixedSize	boolean	determines whether the crop has fixed size even if it is out of image limits.

Crop the image using the given coordinates and size. Returns resource of the cropped image on success or FALSE on failure. Exmpale:

```
// Reads image info.
$pic = new Picture('example.png');
// Crops the image.
$img = $pic->crop(-50, 100, 200, 300, Picture::rgb2int(0xFF, 0, 0xFF, 0.5), false);
// Sends the cropped image to browser.
header('Content-Type: image/png');
imagepng($img);
```

> Note that after the original image is cropped, it is not stored in the file. The changed image exists only in memory.

resize()

```
public function resize(integer $width, integer $height, integer $mode = Picture::PIC_AUTO, integer $maxWidth = null, integer $maxH
```

$width	integer	the desired width of the image. This parameter is ignored if the resizing mode is PIC_AUTO or PIC_AUTOWIDTH.
height	integer	the desired height of the image. This parameter is ignored if the resizing mode is PIC_AUTO or PIC_AUTOWIDTH.
$mode	integer	the resizing mode.
$maxWidth	integer	the upper limit of the width of the resizing image.
$maxHeight	integer	the upper limit of the height of the resizing image.

Resizes an image using the given new width and height. Returns resource of the resized image on success or FALSE on failure.

There are four resizing modes corresponding the appropriate class constants:

- **PIC_MANUAL** - according to this mode the new image will have width and height exactly as the desired **$width** and **$height**.
- **PIC_AUTOWIDTH** - the new image will have height exactly as the desired **$height** but its width will be automatically determined according to

the aspect ratio of the image.
- **PIC_AUTOHEIGHT** - the new image will have width exactly as the desired **$width** but its height will be automatically determined according to the aspect ratio of the image.
- **PIC_AUTO** - according to this mode the desired width and height are fully ignored, but dimensions of the new image will be automatically adjusted to the specified limits: **$maxWidth** and **maxHeight** (if they are defined). The aspect ratio of the image is not changed.

Example:

```
// Reads image info.
$pic = new Picture('example.png');
// Resizes the image.
$img = $pic->resize(0, 0, Utils\Picture::PIC_AUTO, 300, 200);
// Sends the resized image to browser.
header('Content-Type: image/png');
imagepng($img);
```

> Note that after the original image is resized, it is not stored in the file. The changed image exists only in memory.

save()

```
public boolean save(string $filename = null, string $type = null, array $options = null)
```

$filename	string	the new image file. If it is not set the original image file will be used.
$type	string	type of the saving image. Valid values are "png", "jpg", "jpeg", "gif", "wbmp", "xbm", "webp". The default value is "png".
$options	array	array of additional options for different image types.

Saves the changed image to a new file. The method returns TRUE on success or FALSE on failure.

According to **$type** the additional parameter **$options** can have one or more options:

- **quality** - compression level: from 0 (no compression) to 9. This option exists for PNG and JPEG (JPG) files.
- **filters** - allows reducing the PNG file size. It is a bitmask field which may be set to any combination of the PNG_FILTER_XXX constants. PNG_NO_FILTER or PNG_ALL_FILTERS may also be used to respectively disable or activate all filters.
- **foreground** - determines the foreground color. The default foreground color is black. The option exists for XBM and WBMP files.

Example:

```
// Defines the background color.
$bgcolor = Picture::rgb2int(255, 127, 0, 0.5);
// Reads info about the image.
$pic = new Picture('example.jpg');
// Changed the image.
$pic->rotate('45', $bgcolor, true);
$pic->crop(850, -50, 200, 300, $bgcolor);
$pic->resize(0, 0, Utils\Picture::PIC_AUTO, 300, 200);
// Stores the changed image in a new file as a PNG image with maximum compression level.
$pic->save('new.png', 'png', ['quality' => 9]);
```

Protected non-static properties

img

```
protected resource $img
```

The image resource identifier.

info

```
protected array $info = []
```

The image information.

Protected non-static methods

readImageInfo()

```
protected void readImageInfo(string $image)
```

$image	string	specifies the image file you wish to retrieve information about.

Reads information about an image.

getRightSize()

```
protected array getRightSize(integer $mode, integer $dstWidth, integer $dstHeight, integer $srcWidth, integer $srcHeight, integer
```

$mode	integer	the resizing mode.
$dstWidth	integer	the desired width of the resizing image. This parameter is ignored if the resizing mode is PIC_AUTO or PIC_AUTOWIDTH.
$dstHeight	integer	the desired height of the resizing image. This parameter is ignored if the resizing mode is PIC_AUTO or PIC_AUTOHEIGHT.
$srcWidth	integer	real width of the image.
srcHeight	integer	real height of the image.
$maxWidth	integer	the upper limit of the width of the resizing image.
$maxHeight	integer	the upper limit of the height of the resizing image.

Returns new width and height for resizing image according to the specified scale mode. New dimension of the image is returned as a two-element numeric array in which the first element is the width and the second one is the height.

createNewImage()

```
protected resource createNewImage(integer $width, integer $height, integer $bgcolor = null)
```

$width	integer	the width of the new image.
$height	integer	the height of the new image.
$bgcolor	integer	the background color of the new image.

Creates the new true color image of the given size and color.