Resize and process images on the fly with this PHP class

This PHP class helps you to process images. There are two manner you can use this script. One manner is directly from url like (imageprocessor.php?src=image&h=100&w&50&...). This can be used to create thumbnails on the fly.

The second manner is using the class in your website/application code by including the imageprocessor.php in your script.

Thank you for purchasing my script. If you have any questions that are beyond the scope of this help file, please feel free to email me at *wim@sitebase.be*. Thanks so much!

PHP Version

5.x

MySQL

Not needed

Compatible browsers

FireFox 2, FireFox 3, IE6, IE7, IE8, Safari

Using image processor to resize image on the fly

This scripts has two classes, imageprocessor.php and advancedprocessor.php. In this example I'll explain what you can do with imageprocessor.php class. On this page we explain how to use the first manner of using this class. That's by resizing images on the fly with an url.

Resize image with direct url (stretch, fit and crop)

To do this we call the imagprocessor.php script with as parameter

- **src:** for the source image
- **w:** for the new image width
- **q:** for the quality

If you only give in a width or only a height the aspect ratio is preserved.

```
http://[yourserver]/imageprocessor.php?src=images/blue.jpg&w=300&q=100
```

Resize image with direct url and add a watermark

We call the imagprocessor.php script with as parameter src for the source image, w for the new image width and q for the quality. The m parameter is used to select the watermark.

```
http://[yourserver]/imageprocessor.php?src=images/blue.jpg&w=300&q=100&m=watermark.png
```

Resize image with direct url with specific mode

The resize of an image can be done in 3 modes. **Stretch** is the default. And besides the default you can use **fit** or **crop**. In the example below we use the crop mode. You can define the mode by adding the parameter mode in the url.

```
http://[yourserver]/imageprocessor.php?src=images/leaf.jpg&w=300&h=200&q=100&mode=crop
```

Using image processor to re-size image on the fly

Below you find all examples of how to the image processor class in your own code.

Rotate an image using the class

Also rotating images is possible with the Image Processor class. Another advantage of this class is that you can combine actions. In this example I show you how you can combine resize and rotating an image.

```
include(“imageprocessor.php”);
$ImageProcessor = new ImageProcessor();
$ImageProcessor->Load(“images/leaf.jpg”);
$ImageProcessor->Resize(300, 201, RESIZE_STRETCH);
$ImageProcessor->Rotate(90);
$ImageProcessor->Parse(100);
```

You can add this code to a PHP file and then set this PHP file as source for an image.

Resize an image using the class

Resizing an image with this class is as easy as the code below. The last parameter of the Resize method is the type of resizing you want to do. You can chouse between.

- RESIZE_STRETCH
- RESIZE_FIT
- RESIZE_CROP

```
include(“imageprocessor.php”);
$ImageProcessor = new ImageProcessor();
$ImageProcessor->Load(“images/white.jpg”);
$ImageProcessor->Resize(300, 201, RESIZE_STRETCH);
$ImageProcessor->Parse(100);
```

Change the quality of the outputted image

If you want images that load faster, you can change the image quality. This you can do by filling in a number between 0 and 100 as parameters of the Parse methode. Default the parse method uses quality 80.

The lower the number, the lower the quality.

```
include(“imageprocessor.php”);
$ImageProcessor = new ImageProcessor();
$ImageProcessor->Load(“images/white.jpg”);
$ImageProcessor->Resize(300, 201, RESIZE_STRETCH);
$ImageProcessor->Parse(10);
```

Add a watermark to the image with the class

Adding a watermark on an image can be done with the Watermark method. Where the first parameter is the watermark image, second the horizontal position and last the vertical position.

To position the watermark horizontal or vertical you can use pixels but it’s easier to use the special variables.

For the horizontal parameter there are:

- POSITION_LEFT
- POSITION_CENTER
- POSITION_RIGHT

And for the vertical parameter, there are.

- POSITION_BOTTOM
- POSITION_BOTTOM
- POSITION_BOTTOM

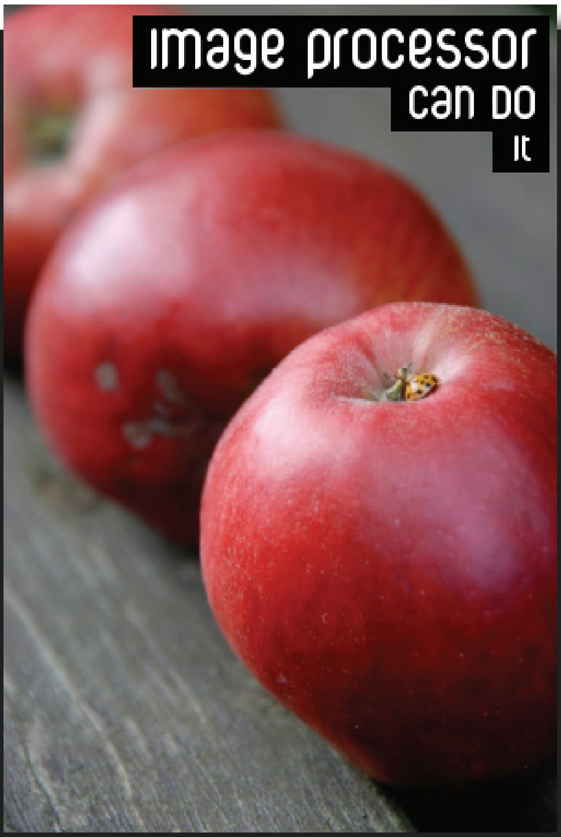
```
include(“imageprocessor.php”);
$ImageProcessor = new ImageProcessor();
$ImageProcessor->Load(“images/brown.jpg”);
$ImageProcessor->Resize(300, null, RESIZE_STRETCH);
$ImageProcessor->Watermark(“watermark.png”, POSITION_RIGHT, POSITION_BOTTOM);
$ImageProcessor->Parse(100);
```

Do custom thing with a resized image resource

With the GetResource method you can get the resize image resource from the class. That is very handy if you want do some extra things with the resized image.

This way you don’t need to save the image first load it back in code and then do your thing with the resized image.

To add the text and the black rectangles to the image we resized the image. Getted the resources from the class. Added the rectangles and the text to the resource and set the resource back to the class. And at the end we parse the image to the screen.



```
<?php
include(“imageprocessor.php”);
$ImageProcessor = new ImageProcessor();
$ImageProcessor->Load(“images/red.jpg”);
$ImageProcessor->Resize(300, null, RESIZE_STRETCH);

// Get resource
$resource = $ImageProcessor->GetResource();

// Add some text to the image
$white = imagecolorallocate($resource, 255, 255, 255);

// Add black rectangles
imagefilledrectangle($resource,70,6,295,44,1);
imagefilledrectangle($resource,210,45,295,68,1);
imagefilledrectangle($resource,265,68,295,90,1);

// Add text
imagettftext($resource, 30, 0, 80, 35, $white, “corpuscare.ttf”, “Image Processor”);
imagettftext($resource, 25, 0, 220, 62, $white, “corpuscare.ttf”, “Can Do”);
imagettftext($resource, 20, 0, 277, 85, $white, “corpuscare.ttf”, “It”);

// Give the resource back to the class
$ImageProcessor->SetResource($resource);

$ImageProcessor->Parse(100);
```

Save image to file

In all the examples above i used the Parse method to display the image on the screen. But ofcourse it’s also possible to save the resized image. This you can do with the Save method.

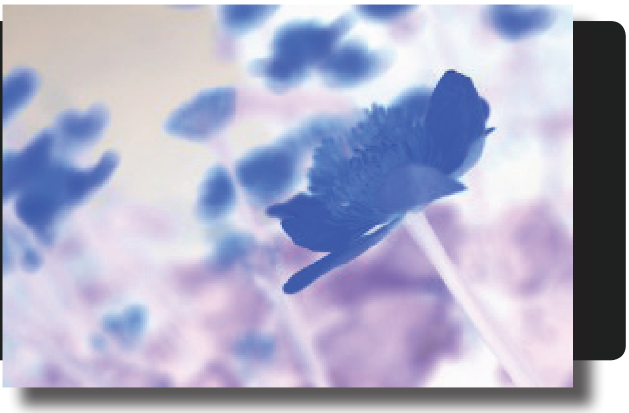
The Save method takes two parameters. The first is the destination file and the second is the image quality.

```
include(“imageprocessor.php”);
$ImageProcessor = new ImageProcessor();
$ImageProcessor->Load(“images/yellow.jpg”);
$ImageProcessor->Resize(300, 201, RESIZE_STRETCH);
$ImageProcessor->Save(“small_yellow.jpg”,90);
```

Add negative effect

You can add a negative effect by using the FilterNegative method.

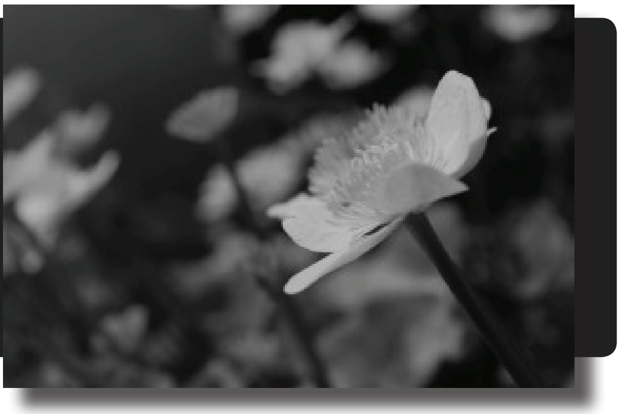
```
include(“imageprocessor.php”);
$ImageProcessor = new ImageProcessor();
$ImageProcessor->Load(“images/yellow.jpg”);
$ImageProcessor->Resize(300, 201, RESIZE_STRETCH);
$ImageProcessor->FilterNegative();
$ImageProcessor->Parse(90);
```



Add grayscale effect

You can add a grayscale effect by using the FilterGray method.

```
include(“imageprocessor.php”);
$ImageProcessor = new ImageProcessor();
$ImageProcessor->Load(“images/yellow.jpg”);
$ImageProcessor->Resize(300, 201, RESIZE_STRETCH);
$ImageProcessor->FilterGray();
$ImageProcessor->Parse(90);
```

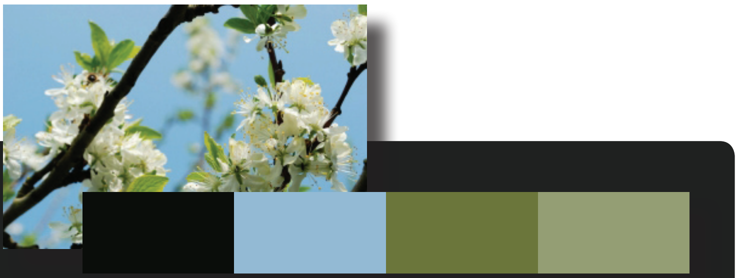


Using advanced image processor class

Besides the ImageProcessor class there’s also the advanced processor class, this is an extension on the imageprocessor class. I splitted this script in two classes so the main class has the won’t get to big and bulky.

Make color map of an image

Create a color scheme from an image.



```
include(“imageprocessor.php”);
include(“advancedprocessor.php”);
$AdvancedProcessor = new AdvancedProcessor();
$map = $AdvancedProcessor->GetColorMap(“images/resize_example.jpg”); $AdvancedProcessor-
>CreateMapImage($map, 75, 40);
```

Create an Photoshop aco file

In previous example I showed you how to make a color scheme from an image. In this example I’ll show you how you can save this color scheme as Phoshop swatch that you can import in Photshop.

```
include(“imageprocessor.php”);
include(“advancedprocessor.php”);
$AdvancedProcessor = new AdvancedProcessor();
$map = $AdvancedProcessor->Gresize_example.jpg”);
$AdvancedProcessor->CreateMapImage($map, 75, 40);
```

Batch resize

It’s also possible to resize a whole directory at once. Below you find the code needed. The batch resize has 5 parameters: source directory, destination directory, width, height, resize mode.

```
include(“imageprocessor.php”);
include(“advancedprocessor.php”);
$AdvancedProcessor = new AdvancedProcessor();
$AdvancedProcessor->BatchResize(“images/”, “thumbs/”, 200, null, RESIZE_CROP);
```