Weck - Image Handler Class

Hello, thank you for purchase!

Short Description

Weck - class which makes working with images easier. It can change image in lot of ways and use cache system to optimize speed. It can handle unlimited number of images in a single cycle. Easy modify code with comments and documentation. Has logging system for recording events or debugging. Compatible with any php framework, can be used in any php system. To use it, create an object of this class, with image configurations. Then use object methods to display uploaded images. You can also use other administration methods to handle with website storages.

Contents

Installation

Upload image_handler.php file, which is located in "class" folder, via FTP client to any place in your project. To know more about using FTP and uploading files fallow this link use FTP Instructions.

To configure work of Image Handler you need to setup some constants inside the class. To do this open uploaded image_handler.php file with any text editor you are comfortable. Now scroll quick reminder instructions to the class declaration. Here you will see two constants CACHE_LOCATION and IMAGES_LOCATION. This two contain path to folders in which class will be store uploaded images and cache files. Write path to this locations in your project.

Notice: path must end with slash. You can use absolute or relative path. To know more about path construction read this article Absolute vs. Relative Paths/Links.

Notice: if you will not configure this two constants, this folders will be created automatically at the image_handler.php file location.

To connect "Image Handler" in your project include image_handler.php above first usage. Example:

```
include_once('path_to_class_folder/image_handler.php');
```

Basic Using

To use Image Handler correct, first you need to create object with settings of output image. Then upload image using object method, and then output image calling other method. You need to create it is own object for each output image variation. Feel free to create as many objects as you need.

Lets suppose you are uploading article thumbnail image and you want it to appear in 600x400 pixels resolution. To do this you need to make three simple steps. Create object, upload image and call image output. This is the example of creation an object with this settings:

```
$args = [
      "width" => 600,
      "height" => 400
      ];
$new_image = new ImageHandler ($args);
```

Notice: if you will not give resolution to function it will apply default values - 100x100 pixels.

To upload image you need to call handleImage method and give it name, type and temporal location of uploaded file. All this values are located in global $_FILES array. Here is the example:

```
$new_image -> handleImage ($_FILES['image']['name'], $_FILES['image']['type'], $_FILES['image']['tmp_name']);
```

Notice: if you will upload image file in the folder, which setup in class as storage for images, with some other way - it will work as well.

To output the image you need to call showImage method and give it name of your image. You can also send image attributes in array with a second parameter. Example:

```
$attributes = [
      'class' => 'test-class',
      'id' => 'test-id',
      'alt' => 'test-"alt"'
];
echo $new_image -> showImage ($_FILES['image']['name'], $attributes);
```

Notice: file must be located in storage folder.

To output the link to image you need to call showLink method and give it name of your image. Example:

```
$link = $new_image -> showLink ($_FILES['image']['name']);
```

Auto Size

In case if you need only one side to be strict sized and a second is adaptive, you can setup width or height to 'auto'. Image Handler will take proportions of the original image and calculate size of second side to keep sides ratio. For example:

```
$args = [
      "width" => 600,
      "height" => 'auto'
];
$new_image = new ImageHandler ($args);
```

Notice: if you will setup both sizes to 'auto' then script will take original sizes of the image.

Dynamic Path

If your source images you are working with located in different folders you can path url to it with image name. For example:

```
$new_image = showImage ('images/p.png');
```

Notice: if you will setup both sizes to 'auto' then script will take original sizes of the image.

Crop

Notice: all size considering cropping must be calculated from original image, not from an output one.

In case if you need to crop an image you can do it by setting up 4 variables. First two X and Y are coordinates in pixels that indicate where top left corner of cropping quadrilateral is begin. Second two is width and height in pixels that indicates size of the cropping quadrilateral. For example:

```
$args = [
      'crop_x' => 100,
      'crop_y' => 100,
      'crop_width' => 600,
      'crop_height' => 500,
      "width" => 600,
      "height" => '500'
];
$new_image = new ImageHandler ($args);
```

Notice: cropping executing before all of other manipulations with image.

Notice: if you will not setup initial coordinates it will use 0 and crop from the beginning of the image. If you will not setup sizes of the cropping area cropping will mot execute.

Notice: if cropping area is not equal in sizes to output dimensions handler will be forced to make second crop while resizing.

Flip

To flip image you need to add an additional parameter 'flip' to settings. Available flip options:

- vertical
- horizontal
- both

For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      'flip' => 'vertical'
];
$new_image = new ImageHandler ($args);
```

Rotate

To rotate image you need to add an additional parameter 'rotate' to settings. Parameter must contain degree in positive numbers. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      'rotate' => 150
];
$new_image = new ImageHandler ($args);
```

Grayscale

To grayscale image you need to add 'grayscale' setting with value 'true'. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      'grayscale' => true
];
$new_image = new ImageHandler ($args);
```

Watermark

To add watermark to your image you need to specify parameters of the watermark when you creating an object. Name of the watermark file, it is position, size, opacity, horizontal and vertical indents from the borders. Available position values:

- top-left
- top-right
- bottom-left
- bottom-right

Size of the thumbnail measure in % from size of the main image. Indent measure in pixels. Here is example:

```
$args = [
      "width" => 600,
      "height" => 400,
      "stamp" => 'thumbnail.jpg',
      "stamp_position" => 'bottom-right',
      "stamp_size" => 10,
      "stamp_opacity" => 10,
      "stamp_indent_horizontal" => 5,
      "stamp_indent_vertical" => 5
];
$new_image = new ImageHandler ($args);
```

Notice: if you will not give this settings to function it will apply default values - position "bottom-right", size 5%, indent 10px, watermark have no default value.

You can upload watermark as simple image, using same methods as when you uploaded main image. Example:

```
$new_watermark = new ImageHandler ();
$new_watermark -> handleImage ($_FILES['watermark']['name'],
$_FILES['watermark']['type'], $_FILES['watermark']['tmp_name']);
```

Notice: you can upload watermark any other way, but it is must be located in storage folder as well.

To put watermark on image use the same image output method as before. Example:

```
echo $new_image -> showImage ($_FILES['image']['name']);
```

Text

To add text to your image you need to specify parameters of the text when you creating an object. Text its self, font file, it is position, size, horizontal and vertical indents from the borders, color and angle. Available position values:

- top-left
- top-right
- bottom-left
- bottom-right

Size of the thumbnail measure in % from size of the main image. Indent measure in pixels. Here is example:

```
$args = [
      "width" => 600,
      "height" => 400,
      "text" => 'Test',
      "text_position" => 'bottom-right',
      "text_font" => 'assets/fonts/roboto/Roboto-Black.ttf',
      "text_size" => 10,
      "text_color" => '#ff5544',
      "text_rotate" => 10,
      "text_indent_horizontal" => 5,
      "text_indent_vertical" => 5
];
$new_image = new ImageHandler ($args);
```

Notice: it is impossible to calculate text width, because letters have different width in different fonts and this data do not available in PHP

script. Width of the text calculating by multiplying font size on count of symbols. If you need to attach text to the right corner use negative number for horizontal indent. Sorry for inconvenience!

Notice: if you will not give this settings to function it will apply default values - position "bottom-right", size 5%, indent 10px, font size 14px, font color white, font have no default value.

Notice: fonts must be stored locally, no remote urls.

Opacity

To add opacity to your image you need to add an additional parameter 'opacity' to settings. Parameter must contain number in % format, where 100% is no transparency and 0 is fully transparent. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
       "opacity" => 54
];
$new_image = new ImageHandler ($args);
```

Brightness

To correct image brightness add 'brightness' to settings. Parameter must contain number that will indicate brightness level. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      "brightness" => 50
];
$new_image = new ImageHandler ($args);
```

Contrast

To correct image contrast add 'contrast' to settings. Parameter must contain number that will indicate brightness level. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      "contrast" => 50
];
$new_image = new ImageHandler ($args);
```

Frame

To add frame to your image just add 'frame' parameter with a name of the frame image. Just make shore this image is transparent and has right proportions. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      "frame" => 'frame-image.png'
];
$new_image = new ImageHandler ($args);
```

Background

To add background color to your image you need to add an additional parameter 'background_color' to settings. Parameter must contain color code in '#ff3445' format. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      "background_color" => '#F95'
];
$new_image = new ImageHandler ($args);
```

Edges

To to spotlight edges on your image you need to add 'edges' setting with value 'true'. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      'edges' => true
];
$new_image = new ImageHandler ($args);
```

Reverse Colors

To to reverse colors on your image you need to add 'reverse' setting with value 'true'. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      'reverse' => true
];
$new_image = new ImageHandler ($args);
```

Emboss

To to apply emboss filter on your image you need to add 'emboss' setting with value 'true'. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      'emboss' => true
];
$new_image = new ImageHandler ($args);
```

Effects Order

You can rearrange order in which modifications are applying to original image. To do this just add setting 'order' - array with the names of the effects you can arrange, and move the elements of the array in order that you need. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
// ... different effects here ...
      'order' => [
            "grayscale",
            "brightness",
            "contrast",
            "edges",
            "emboss",
            "reverse",
            "opacity",
            "rotate",
            "flip",
            "frame",
            "stamp",
            "text"
      ]
];
$new_image = new ImageHandler ($args);
```

Notice: do not delete, add or modify elements of the order, if you do handler will ignore it and apply effects in default order.

Converter

You can setup a format of output image by adding an additional parameter 'format' to settings. Parameter must contain 'png' or 'jpg' string, PNG is a default output format. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      'format' => 'png'
];
$new_image = new ImageHandler ($args);
```

Notice: JPEG format do not have transparency option.

Compressor

You can compress output JPEG image by setting up quality parameter, it mast contain % value from 1 to 100. For example:

```
$args = [
      "width" => 600,
      "height" => 400,
      'format' => 'jpg'
      'jpeg_quality' => 90
];
$new_image = new ImageHandler ($args);
```

Notice: JPEG format do not have transparency option.

Managing

Available managing options:

- Render table of the images storage containment
- Render table of the cache storage containment
- Clear cache storage folder
- Render list of log events
- Clear log

All this methods will return string with request, data packed in HTML string. You can setup the HTML class for the content container via first function parameter.

Render table of the images storage containment:

```
echo ImageHandler :: showImagesList("class");
```

Render table of the cache storage containment:

```
echo ImageHandler :: showCacheList("class");
```

Clear cache storage folder:

```
ImageHandler :: deleteCache();
```

Logging

You can activate logging system to debug or monitor images events on your site. It use next marks of value:

- Error - wrong functioning of some method
- Warning - event which require your attention
- Notice - event about which you may be interesting to know
- Info - simple event
- Debug - event which require code debugging

To activate the log you need to set ACTIVATE_LOG constant in 'true' position. It is located at the beginning of the class declaration. Also you can assign your own path and name of the log file in the LOG_FILE constant. Example:

```
const ACTIVATE_LOG = true;
const LOG_FILE     = '/logs/images_log.txt';
```

Managing log. Render list of log events:

```
echo ImageHandler :: showLog("class");
```

Clear log:

```
ImageHandler :: deleteLog();
```

Get a Free Gift

You can get a free gift if you will contact me and share your experience of using this item. Your story may help to improve item quality, and you will get this improvements for free with the next update. Dependently from use of your experience I will define "size" of the gift. I can send you any item from my library for free, give you free freelance hours or more. You can contact me via Envato profile page or use contacts from this documentation, it is located in the blue sidebar under the navigation. Thank you for purchase and may the Force be with you!

Contacts

Use this links to contact me:

- Envato
- Email
- Telegram