

# verot.net - frequently asked questions

[php\\_class\\_upload\\_faq.htm](#)

## I have a problem, can you help me?

**Yes, but first you need to check whether you are using the [latest version of the class](#).** Then, you should [search in the site](#), maybe someone already solved your issue. Then, you can post in the [forums](#), with as much details as possible. You can also [contact me by email](#), and again, with as much details as possible.

With your question, always include the relevant part of your code, and the log produced by the class. To obtain the code, just add this line at the end of your code:

```
echo $handle->log;
```

## Uploading and image manipulation works fine, but fails on large or big images (blank screen)

**You are probably running out of memory. When you resize, crop, convert an image, the actual size in KB of the picture does not matter. Indeed, the picture is converted into a bitmap in memory, so what matters is the number of pixels. Roughly, you need to multiply the number of pixels by 8 to have an estimate of the memory required for the image.**

**To process large images, you need to raise the PHP memory limit, usually in php.ini. On shared hosting, you probably can't edit your php.ini, so you can try the following, although it will work only if the hosting company allows it, which they usually don't.**

**You can add this in your script:**

```
ini_set ( "memory_limit", "40M" )
```

Or in a .htaccess file:

```
php_value memory_limit 40M
```

If the above doesn't work, ask your hosting company.

## I have an error No JPEG create support when I upload an image, even if GD is installed properly on my server?

**It is likely that the web server doesn't have the right permissions to create a file in the directory you upload the files in. If you upload your files in /home/user/foo/, change the permissions on the foo directory. You can do this via FTP, or via the command line. Try 777 first and the class should upload the files properly. Then reduce the permissions to something like 755. Note: from v0.20, directories can be created or chmod' automatically; also, more explanatory error messages are generated.**

## I have an error open\_basedir restriction in place? when I upload a file?

**You probably have a an *open\_basedir* restriction in your php.ini, or you run in a *chroot* environment. Try to add the upload path of PHP in the *open\_basedir*. The class can work with *open\_basedir*, but a few features will not be available: the image properties will not be available before calling \$foo->process()**

## My code seems to be fine, it work on one server, but fails on another?

**Likely your PHP setup is different. You may need to raise the PHP memory limit, upload limit, or script execution time limit, usually in *php.ini*:**

```
php_value memory_limit = 40M
upload_max_filesize = 30M
```

```
post_max_size = 30M
max_execution_time = 60
```

On shared hosting, you probably can't edit your *php.ini*, so you can try the following methods, although it will work only if the hosting company allows it, which they usually don't.

you can add this in your script:

```
ini_set ( "memory_limit", "40M")
```

Or in a .htaccess file:

```
php_value memory_limit 40M
```

### **Can the class work with Flash uploaders?**

Flash has the bad habit of setting the MIME type to application/octet-stream for any file. Versions of the class before version 0.26 could not reliably deal with files uploaded from Flash. From version 0.26, the class has improved MIME detection and if the detection fails, the class guesses the MIME type from the file extension. So the class will work out of the box with Flash uploaders.

### **How do I get the dimensions of an image before I convert or resize it?**

Before calling `$foo->process()` (and if you don't have *open\_basedir* restrictions in place), you can read some class variables to obtain this information:

`$foo->file_is_image`, `$foo->image_src_x`, `$foo->image_src_y`, `$foo->image_src_pixels`,  
`$foo->image_src_type` and `$foo->image_src_bits`

### **How do I get the dimensions of an image after I have converted or resized it?**

After calling `$foo->process()`, you can read some class variables to obtain this information:

`$foo->image_dst_x` and `$foo->image_dst_y`

### **When uploading a file with `file_auto_rename` set to TRUE, how do I retrieve the new filename?**

You can read some class variables after calling `$foo->process()`. The following ones can be used to get the filename details:

`$foo->file_dst_name`, `$foo->file_dst_name_body`, `$foo->file_dst_name_ext`, `$foo->file_dst_path` and `$foo->file_dst_pathname`

### **What if I want to upload 1 image and resize it twice to create 2 images: a small thumbnail, and a large image (but with a max size)?**

You can call `Process()` several times. So the first `Process()` will for instance upload the image as-is, and the second one upload the image as a thumbnail. Example:

1. new `Upload(...)` (only once!)
2. set parameters such as `image_resize`, etc...  
call `Process()` -> first image created
3. set parameters such as `image_resize`, with different sizes, or convert it...  
call `Process()` -> second image created
4. etc...
5. call `Clean()` when you have created all the images you needed.

### **How do I overwrite again and again the same file?**

You should use `$handle->file_auto_rename = false;` to prevent the class renaming the file

automatically, and then `$handle->file_overwrite = true;` to allow the class to overwrite the file. This also works when working on local files.

### What about multiple uploads?

The class itself handles one uploaded file at a time. In case of multiple uploads, the `$_FILES` array has a different structure, so you need first to re-structure it so it can be looped through, and each element of the array used to instantiate the class.

For instance, you can do:

```
$files = array();
foreach ($_FILES['my_field'] as $k => $l) {
    foreach ($l as $i => $v) {
        if (!array_key_exists($i, $files))
            $files[$i] = array();
        $files[$i][$k] = $v;
    }
}
```

and then:

```
foreach ($files as $file) {
    $handle = new Upload($file);
    if ($handle->uploaded) {
        $handle->Process("./");
        if ($handle->processed) {
            echo 'OK';
        } else {
            echo 'Error: ' . $handle->error;
        }
    } else {
        echo 'Error: ' . $handle->error;
    }
    unset($handle);
}
```

**My code looks OK, however the server doesn't seem to receive any images?**

**Make sure that you set the enctype property of your form tag, as following:**

**enctype="multipart/form-data"**

**In some cases, users have reported that the hidden field `MAX_FILE_SIZE` should be set.**

**Is this class compatible PHP4 and PHP5?**

**Yes.**

**Is there any way to debug the class?**

**You can display the log (HTML formatted) after processing some uploads. Output it like this:**

**echo \$foo->log;**

**Can the class manipulate BMP images?**

**No, the class can only process (resize, convert...) JPG, PNG and GIF images, depending on your configuration of GD. However, you can upload BMP images as any other files.**

**Update:** [version 0.25](#) can handle BMP files.

**Is this class free?**

**The class.upload.php is released as GPL. It means that you can use it freely in any GPL project. If you wish to use it in a closed source project, then contact me for a [commercial license](#).**

**Is this class supported, or actively maintained?**

**The class is actively maintained, but on my free time. You can help me spending more time on it with [a donation](#).**

I provide a limited support by email. If you wish a more comprehensive support, please [contact me](#)