php

- Downloads
- Documentation
- Get Involved
- Help

Search

Keyboard Shortcuts

?	This help
j	Next menu item
k	Previous menu item
g p	Previous man page
g n	Next man page
G	Scroll to bottom
g g	Scroll to top
g h	Goto homepage
g s	

Goto search
(current page)
/
Focus search box

getimagesizefromstring »
« gd_info

- PHP Manual
- Function Reference
- Image Processing and Generation
- GD
- GD and Image Functions

Change language: English

Edit Report a Bug

getimagesize

(PHP 4, PHP 5, PHP 7)

getimagesize — Get the size of an image

Description ¶

array **getimagesize** (string `$filename` [, array `&$imageinfo`])

The **getimagesize()** function will determine the size of any supported given image file and return the dimensions along with the file type and a *height/width* text string to be used inside a normal HTML `IMG` tag and the correspondent HTTP content type.

getimagesize() can also return some more information in `imageinfo` parameter.

Caution

This function expects `filename` to be a valid image file. If a non-image file is supplied, it may be incorrectly detected as an image and the function will return successfully, but the array may contain nonsensical values.

Do not use **getimagesize()** to check that a given file is a valid image. Use a purpose-built solution such as the Fileinfo extension instead.

> **Note**: Note that JPC and JP2 are capable of having components with different bit depths. In this case, the value for "bits" is the highest bit depth encountered. Also, JP2 files may contain *multiple JPEG 2000 codestreams*. In this case, **getimagesize()** returns the values for the first codestream it encounters in the root of the file.

> **Note**: The information about icons are retrieved from the icon with the highest bitrate.

Parameters ¶

`filename`

> This parameter specifies the file you wish to retrieve information about. It can reference a local file or (configuration permitting) a remote file using one of the supported streams.

`imageinfo`

> This optional parameter allows you to extract some extended information from the image file. Currently, this will return the different JPG APP markers as an associative array. Some programs use these APP markers to embed text information in images. A very common one is to embed » IPTC information in the APP13 marker. You can use the iptcparse() function to parse the binary APP13

marker into something readable.

Return Values ¶

Returns an array with up to 7 elements. Not all image types will include the *channels* and *bits* elements.

Index 0 and 1 contains respectively the width and the height of the image.

> **Note**:
>
> Some formats may contain no image or may contain multiple images. In these cases, **getimagesize()** might not be able to properly determine the image size. **getimagesize()** will return zero for width and height in these cases.

Index 2 is one of the IMAGETYPE_XXX constants indicating the type of the image.

Index 3 is a text string with the correct *height="yyy" width="xxx"* string that can be used directly in an IMG tag.

mime is the correspondant MIME type of the image. This information can be used to deliver images with the correct HTTP *Content-type* header:

Example #1 getimagesize() and MIME types

```php
<?php
$size = getimagesize($filename);
$fp = fopen($filename, "rb");
if ($size && $fp) {
    header("Content-type: {$size['mime']}");
    fpassthru($fp);
    exit;
} else {
    // error
}
?>
```

channels will be 3 for RGB pictures and 4 for CMYK pictures.

bits is the number of bits for each color.

For some image types, the presence of *channels* and *bits* values can be a bit confusing. As an example, GIF always uses 3 channels per pixel, but the number of bits per pixel cannot be calculated for an animated GIF with a global color table.

On failure, `FALSE` is returned.

Errors/Exceptions ¶

If accessing the `filename` image is impossible **getimagesize()** will generate an error of level `E_WARNING`. On read error, **getimagesize()** will generate an error of level `E_NOTICE`.

Changelog ¶

Version	Description
7.1.0	Added WebP support.
5.3.0	Added icon support.
5.2.3	Read errors generated by this function downgraded to `E_NOTICE` from `E_WARNING`.
4.3.2	Support for JPC, JP2, JPX, JB2, XBM, and WBMP became available.
4.3.2	JPEG 2000 support was added for the `imageinfo` parameter.

Version	Description
4.3.0	*bits* and *channels* are present for other image types, too.
4.3.0	Support for SWC and IFF was added.
4.2.0	Support for TIFF was added.
4.0.6	Support for BMP and PSD was added.

Examples ¶

Example #2 getimagesize() example

```
<?php
list($width, $height, $type, $attr) = getimagesize("img/flag.jpg");
echo "<img src=\"img/flag.jpg\" $attr alt=\"getimagesize() example\" />";
?>
```

Example #3 getimagesize (URL)

```
<?php
$size = getimagesize("http://www.example.com/gifs/logo.gif");

// if the file name has space in it, encode it properly
$size = getimagesize("http://www.example.com/gifs/lo%20go.gif");

?>
```

Example #4 getimagesize() returning IPTC

```
<?php
$size = getimagesize("testimg.jpg", $info);
if (isset($info["APP13"])) {
    $iptc = iptcparse($info["APP13"]);
    var_dump($iptc);
}
?>
```

Notes ¶

Note:

This function does not require the GD image library.

See Also ¶

- image_type_to_mime_type() - Get Mime-Type for image-type returned by getimagesize, exif_read_data, exif_thumbnail, exif_imagetype
- exif_imagetype() - Determine the type of an image
- exif_read_data() - Reads the EXIF headers from an image file
- exif_thumbnail() - Retrieve the embedded thumbnail of an image

add a note

User Contributed Notes 33 notes

up

down
51
james dot relyea at zifiniti dot com ¶
8 years ago

```
As noted below, getimagesize will download the entire image before it checks for the requested information. This is extremely slow on large images that are accessed
remotely. Since the width/height is in the first few bytes of the file, there is no need to download the entire file. I wrote a function to get the size of a JPEG by
streaming bytes until the proper data is found to report the width and height:

<?php
// Retrieve JPEG width and height without downloading/reading entire image.
function getjpegsize($img_loc) {
    $handle = fopen($img_loc, "rb") or die("Invalid file stream.");
    $new_block = NULL;
    if(!feof($handle)) {
        $new_block = fread($handle, 32);
        $i = 0;
        if($new_block[$i]=="\xFF" && $new_block[$i+1]=="\xD8" && $new_block[$i+2]=="\xFF" && $new_block[$i+3]=="\xE0") {
            $i += 4;
            if($new_block[$i+2]=="\x4A" && $new_block[$i+3]=="\x46" && $new_block[$i+4]=="\x49" && $new_block[$i+5]=="\x46" && $new_block[$i+6]=="\x00") {
                // Read block size and skip ahead to begin cycling through blocks in search of SOF marker
                $block_size = unpack("H*", $new_block[$i] . $new_block[$i+1]);
                $block_size = hexdec($block_size[1]);
                while(!feof($handle)) {
                    $i += $block_size;
                    $new_block .= fread($handle, $block_size);
                    if($new_block[$i]=="\xFF") {
                        // New block detected, check for SOF marker
                        $sof_marker = array("\xC0", "\xC1", "\xC2", "\xC3", "\xC5", "\xC6", "\xC7", "\xC8", "\xC9", "\xCA", "\xCB", "\xCD", "\xCE", "\xCF");
                        if(in_array($new_block[$i+1], $sof_marker)) {
                            // SOF marker detected. Width and height information is contained in bytes 4-7 after this byte.
                            $size_data = $new_block[$i+2] . $new_block[$i+3] . $new_block[$i+4] . $new_block[$i+5] . $new_block[$i+6] . $new_block[$i+7] .
$new_block[$i+8];
                            $unpacked = unpack("H*", $size_data);
                            $unpacked = $unpacked[1];
                            $height = hexdec($unpacked[6] . $unpacked[7] . $unpacked[8] . $unpacked[9]);
                            $width = hexdec($unpacked[10] . $unpacked[11] . $unpacked[12] . $unpacked[13]);
                            return array($width, $height);
                        } else {
                            // Skip block marker and read block size
                            $i += 2;
                            $block_size = unpack("H*", $new_block[$i] . $new_block[$i+1]);
                            $block_size = hexdec($block_size[1]);
                        }
                    } else {
                        return FALSE;
                    }
                }
            }
        }
    }
    return FALSE;
}
?>
```

up
down

3
simon dot waters at surevine dot com ¶
2 years ago

```
Note: getimage size doesn't attempt to validate image file formats

It is possible for malformed GIF images to contain PHP and still have valid dimensions.

Programmers need to ensure such images are validated by other tools, or never treated as PHP or other executable types (enforcing appropriate extensions, avoiding user controlled renaming, restricting uploaded images to areas of the website where PHP is not enabled).

http://ha.ckers.org/blog/20070604/passing-malicious-php-through-getimagesize/
```

up
down
16
kumarldh at gmail dot com ¶
6 years ago

```
I spent quite a time and realised one needs "allow_url_fopen" turned on to be able to use getimagesize(). Hope this help others.
```

up
down
9
tomasz at trejderowski dot pl ¶
4 years ago

```
If you want to "convert" value returned by "getimagesize()" as index "2" into something more human-readable, you may consider using a function like this one:

    $imageTypeArray = array
    (
        0=>'UNKNOWN',
        1=>'GIF',
        2=>'JPEG',
        3=>'PNG',
        4=>'SWF',
        5=>'PSD',
        6=>'BMP',
        7=>'TIFF_II',
        8=>'TIFF_MM',
        9=>'JPC',
        10=>'JP2',
        11=>'JPX',
        12=>'JB2',
        13=>'SWC',
        14=>'IFF',
        15=>'WBMP',
        16=>'XBM',
        17=>'ICO',
        18=>'COUNT'
    );

    $size = getimagesize($filename);

    $size[2] = $imageTypeArray[$size[2]];

Or something similar.
```

up
down
13

php dot net at dannysauer dot com ¶

12 years ago

```
Note that, if you're going to be a good programmer and use named constatnts (IMAGETYPE_JPEG) rather than their values (2), you want to use the IMAGETYPE variants -
IMAGETYPE_JPEG, IMAGETYPE GIF, IMAGETYPE_PNG, etc.  For some reason, somebody made a horrible decision, and IMG_PNG is actually 4 in my version of PHP, while
IMAGETYPE_PNG is 3.  It took me a while to figure out why comparing the type against IMG_PNG was failing...
```

up
down
4

kazuya ¶

3 years ago

```
i made function img_resize($path,$tmp_name,$new_name,$new_width)
this could be useful.

<?php

$new_file = img_resize("./img/", "test.jpg","copy_test.jpg",300);
echo "<IMG src = '$new_file'>";

function img_resize($path,$tmp_name,$new_name,$new_width){
    if (!file_exists($path.$filename)){
        echo "file not found!";
        exit;
    }
    if (!is_writable($path)){
        echo "error:permission denied!";
        exit;
    }
    list($width, $height) = getimagesize($path . $tmp_name);
    $new_height = abs($new_width * $height / $width);
    $image_p = imagecreatetruecolor($new_width, $new_height);
    $image = imagecreatefromjpeg($path . $tmp_name);
    imagecopyresampled($image_p, $image, 0, 0, 0, 0,
                        $new_width, $new_height, $width, $height);
    imagejpeg($image_p, $path . $new_name);
    return $path.$new_name;
}

?>
```

up
down
3

@hill79 ¶

7 years ago

```
I needed to find image dimensions for use in some dynamic css, since getimagesize() returns width="x" height="y" at index 3 I had to convert that to a valid CSS
format.

Thought I'd share the function in case anyone else needs the same.

<?php
function cssifysize($img) {
$dimensions = getimagesize($img);
$dimensions = str_replace("=\"", ":", $dimensions['3']);
$dimensions = str_replace("\"", "px;", $dimensions);
return $dimensions;
}
```

```
returns width:x;height:y;
?>
```

I expect there's a way of making that more efficient
up
down
3
geoff at spacevs dot com ¶
7 years ago
This function returns the width and height of a JPEG image from a string, allowing the dimensions of images stored in a database to be retrieved without writing them to the disk first, or using "imagecreatefromstring" which is very slow in comparison.

```
<?PHP
function getJPEGImageXY($data) {
        $soi = unpack('nmagic/nmarker', $data);
        if ($soi['magic'] != 0xFFD8) return false;
        $marker = $soi['marker'];
        $data   = substr($data, 4);
        $done   = false;

        while(1) {
                if (strlen($data) === 0) return false;
                switch($marker) {
                        case 0xFFC0:
                                $info = unpack('nlength/Cprecision/nY/nX', $data);
                                return array($info['X'], $info['Y']);
                                break;

                        default:
                                $info   = unpack('nlength', $data);
                                $data   = substr($data, $info['length']);
                                $info   = unpack('nmarker', $data);
                                $marker = $info['marker'];
                                $data   = substr($data, 2);
                                break;
                }
        }
}
?>
```

Doing this 10,000 times takes 0.43 seconds, compared with using imagecreatefromstring/imagesx/imagesy which takes around 1.52 seconds to do the same.

Do not use this instead of getimagesize when dealing with files, getimagesize is much faster coming in at 0.15 seconds.
up
down
2
Urchin ¶
6 years ago
On a Debian machine I had a lot of Notices in my log because the system did not understand spaces. While the str_replace and rawurlencode options are ok for remote images, for the local system it is of no use.

I used the following:

```
getimagesize('"'.$location.'"');
```

so basically I quoted the location (single_quot-dubble_quote-single_quote).
up
down
2
webmaster AT theparadox DOT org ¶
14 years ago
I figured others have wanted to scale an image to a particular height or width while preserving the height/width ratio. So here are the functions I wrote to accomplish this. Hopefully they'll save somebody else the five minutes it took to write these.

You give the filename and the dimension you want to use, and these functions return the opposite dimension:

```
function scale_to_height ($filename, $targetheight) {
   $size = getimagesize($filename);
   $targetwidth = $targetheight * ($size[0] / $size[1]);
   return $targetwidth;
}

function scale_to_width ($filename, $targetwidth) {
   $size = getimagesize($filename);
   $targetheight = $targetwidth * ($size[1] / $size[0]);
   return $targetheight;
}
```
up
down
5
Jesus Zamora ¶
6 years ago
Returns a array with 4 elements.
The 0 index is the width of the image in pixels.
The 1 index is the height of the image in pixels.
The 2 index is a flag for the image type:

1 = GIF, 2 = JPG, 3 = PNG, 4 = SWF, 5 = PSD, 6 = BMP, 7 = TIFF(orden de bytes intel), 8 = TIFF(orden de bytes motorola), 9 = JPC, 10 = JP2, 11 = JPX, 12 = JB2, 13 = SWC, 14 = IFF, 15 = WBMP, 16 = XBM.

The 3 index contains ' height="yyy" width="xxx" '
up
down
2
Coodiss at w3bbix dot net ¶
12 years ago
Heres a easy way to scale images to the <td> that they are in
*this is broken up so anyone can understand it :)

```
<?
$imageinfo = getimagesize("images/picture.jpg");

$ix=$imageinfo[0];
$iy=$imageinfo[1];

$widthscale = $ix/175;  //<TD> WIDTH
$heightscale = $iy/175; //<TD> HEIGHT

if($widthscale < 1)
```

```
$nwidth = $ix*$widthscale;
else
$nwidth = $ix/$widthscale;

if($heightscale < 1)
$nheight = $iy*$heightscale;
else
$nheight = $iy/$heightscale;

?>
```

up
down
3
alexyam at live dot com ¶
5 years ago

```
I wanted to use getimagesize() on .SWF files stored in the database as blob data and couldn't find a simple solution, so I created my own.

I am releasing this code under the MIT license to save everyone some time:

<?php
/*
    ----------------------------------------------------------------------
    PHP Blob Data As File Stream v1.0 (C) 2012 Alex Yam <alexyam@live.com>
    This code is released under the MIT License.
    ----------------------------------------------------------------------
    [Summary]

    A simple class for PHP functions to read and write blob data as a file
    using a stream wrapper.

    Particularly useful for running getimagesize() to get the width and
    height of .SWF Flash files that are stored in the database as blob data.

    Tested on PHP 5.3.10.

    ----------------------------------------------------------------------
    [Usage Example]

    //Include
        include('./blob_data_as_file_stream.php');

    //Register the stream wrapper
        stream_wrapper_register("BlobDataAsFileStream", "blob_data_as_file_stream");

    //Fetch a .SWF file from the Adobe website and store it into a variable.
    //Replace this with your own fetch-swf-blob-data-from-database code.
        $swf_url = 'http://www.adobe.com/swf/software/flash/about/flashAbout_info_small.swf';
        $swf_blob_data = file_get_contents($swf_url);

    //Store $swf_blob_data to the data stream
        blob_data_as_file_stream::$blob_data_stream = $swf_blob_data;

    //Run getimagesize() on the data stream
        $swf_info = getimagesize('BlobDataAsFileStream://');
        var_dump($swf_info);
    ----------------------------------------------------------------------
    [Usage Output]

    array(5) {
      [0]=>
      int(159)
      [1]=>
      int(91)
      [2]=>
      int(13)
      [3]=>
      string(23) "width="159" height="91""
      ["mime"]=>
      string(29) "application/x-shockwave-flash"
    }

*/

class blob_data_as_file_stream {

    private static $blob_data_position = 0;
    public static $blob_data_stream = '';

    public static function stream_open($path,$mode,$options,&$opened_path){
        static::$blob_data_position = 0;
        return true;
    }

    public static function stream_seek($seek_offset,$seek_whence){
        $blob_data_length = strlen(static::$blob_data_stream);
        switch ($seek_whence) {
            case SEEK_SET:
                $new_blob_data_position = $seek_offset;
                break;
            case SEEK_CUR:
                $new_blob_data_position = static::$blob_data_position+$seek_offset;
                break;
            case SEEK_END:
                $new_blob_data_position = $blob_data_length+$seek_offset;
                break;
            default:
                return false;
        }
        if (($new_blob_data_position >= 0) AND ($new_blob_data_position <= $blob_data_length)){
            static::$blob_data_position = $new_blob_data_position;
            return true;
        }else{
            return false;
        }
    }

    public static function stream_tell(){
        return static::$blob_data_position;
    }
    public static function stream_read($read_buffer_size){
        $read_data = substr(static::$blob_data_stream,static::$blob_data_position,$read_buffer_size);
        static::$blob_data_position += strlen($read_data);
        return $read_data;
    }

    public static function stream_write($write_data){
        $write_data_length=strlen($write_data);
        static::$blob_data_stream = substr(static::$blob_data_stream,0,static::$blob_data_position).
            $write_data.substr(static::$blob_data_stream,static::$blob_data_position+=$write_data_length);
        return $write_data_length;
    }

    public static function stream_eof(){
        return static::$blob_data_position >= strlen(static::$blob_data_stream);
    }

}
?>
```

up
down
4
info at alex-lawrence dot com ¶
9 years ago

```
Could be useful (didn´t know where to post it):

function getImageErrors( $filename, $type = "", $minWidth = 0, $minHeight = 0, $maxWidth = 0, $maxHeight = 0, $maxFileSize = 0 )
{
    $errors = array();
    if ( file_exists( $filename ) )
    {
        $ending = substr( $filename, strpos( $filename, "." ) );
        if ( is_array( $type ) )
        {
            $isTypeOf = false;
            foreach( $type as $eachtype )
            {
                if ( $ending == $eachtype )
                {
                    $isTypeOf = true;
                }
            }
            if ( ! $isTypeOf )
            {
                $errors[ 'type' ] = $ending;
            }
        }
        elseif ( $type != "" )
        {
            if ( $ending != $type )
            {
                $errors[ 'type' ] = $ending;
            }
        }
        $size = getimagesize( $filename );
        if ( $size[ 0 ] < $minWidth )
        {
            $errors[ 'minWidth' ] = $size[ 0 ];
        }
        if ( $size[ 1 ] < $minHeight )
        {
            $errors[ 'minHeight' ] = $size[ 1 ];
        }
        if ( ( $maxWidth > $minWidth ) && ( $size[ 0 ] > $maxWidth ) )
        {
            $errors[ 'maxWidth' ] = $size[ 0 ];
        }
        if ( ( $maxHeight > $minHeight ) && ( $size[ 1 ] > $maxHeight ) )
        {
            $errors[ 'maxHeight' ] = $size[ 1 ];
        }
        if ( ( $maxFileSize > 0 ) && ( filesize( $filename ) > $maxFileSize ) )
        {
            $errors[ 'maxFileSize' ] = filesize( $filename );
        }
    }
    else
    {
        $errors[ 'filename' ] = "not existing";
    }
    return ( count( $errors ) > 0 ? $errors : null );
}
```

up
down
1
cloned at clonedmadman dot com ¶
9 years ago

Well, I am making a script which will resize the image when uploaded, however, i am making a multi-uploader, so i came across with a problem: an efficient way of getting a pictures height and width and storing them in an array to resize later. This is what i came up with:

```
<?php
$links = array("test1.jpg", "test2.png");
$sizearray = array();
$count = count($links);
for($i = 0; $i < $count; $i++) {
    $size = getimagesize($links[$i]);
    list($width, $height) = $size;
    $sizearray[$links[$i]] = array("width" => $width, "height" => $height);
}
print_r($sizearray);
// which will print out: Array ( [test1.jpg] => Array ( [width] => 300 [height] => 400 ) [test2.png] => Array ( [width] => 680 [height] => 100 ) )
?>
```

up
down
1
pfarthing at hotmail dot com ¶
9 years ago

Correction: to find $y2 it should be...

```
// set y side to a proportional size
$y2 = $m * $x_max; // not $x1

Thanks Norbert =)
```

up
down
1
info at personalmis dot com ¶
9 years ago

```
Seems the various ways people are trying to proportionaly scale an image, up or down, could be more straight forward if one remembers ones algebra.

The formula is, y = mx, where m is the slope of the line. This is the ratio of y:x or m = y/x.

So if...

// max values for x and y
$y_max = 600;
$x_max = 800;

// image size
$y1 = 2000;
$x1 = 3000;

// use width for scaling
if ($x1 > $x_max)
{
    // find slope
    $m = $y1/$x1;
    // set x side to max
    $x2 = $x_max;
    // set y side to a proportional size
    $y2 = $m * $x1;
}

The new image proportionally scaled will be x2 = 800, y2 = 533 (rounded).

To do it from the y side, simply reverse the x's and y's.
```

up
down
1
ajreading at classixshop dot com ¶
12 years ago

```
A simple piece of code i wrote to proportionally resize an image to a max height and width then display it

<?php
// Max height and width
$max_width = 100;
$max_height = 100;

// Path to your jpeg

$upfile '/path/to/file.jpg';
    Header("Content-type: image/jpeg");

    $size = GetImageSize($upfile); // Read the size
```

```
            $width = $size[0];
            $height = $size[1];

            // Proportionally resize the image to the
            // max sizes specified above

            $x_ratio = $max_width / $width;
            $y_ratio = $max_height / $height;

            if( ($width <= $max_width) && ($height <= $max_height) )
            {
                 $tn_width = $width;
                 $tn_height = $height;
            }
            elseif (($x_ratio * $height) < $max_height)
            {
                 $tn_height = ceil($x_ratio * $height);
                 $tn_width = $max_width;
            }
            else
            {
                 $tn_width = ceil($y_ratio * $width);
                 $tn_height = $max_height;
            }
       // Increase memory limit to support larger files

       ini_set('memory_limit', '32M');

       // Create the new image!
       $src = ImageCreateFromJpeg($upfile);
       $dst = ImageCreateTrueColor($tn_width, $tn_height);
       ImageCopyResized($dst, $src, 0, 0, 0, 0, $tn_width, $tn_height, $width, $height);
       ImageJpeg($dst);
  // Destroy the images
  ImageDestroy($src);
  ImageDestroy($dst);
  ?>
```

up
down
1
mail at soylentgreens dot com ¶
12 years ago

```
How about this for cropping images...

<?php

$imgfile = "img.jpg";
$cropStartX = 300;
$cropStartY = 250;
$cropW   = 200;
$cropH   = 200;

// Create two images
$origimg = imagecreatefromjpeg($imgfile);
$cropimg = imagecreatetruecolor($cropW,$cropH);
```

```
// Get the original size
list($width, $height) = getimagesize($imgfile);

// Crop
imagecopyresized($cropimg, $origimg, 0, 0, $cropStartX, $cropStartY, $width, $height, $width, $height);

// TODO: write code to save new image
// or, just display it like this:
header("Content-type: image/jpeg");
imagejpeg($cropimg);

// destroy the images
imagedestroy($cropimg);
imagedestroy($origimg);

?>
```

up
down
2
Steve ¶
6 years ago

```
The list of defined IMAGETYPE_ constants is on the manual page for exif_imagetype:

http://www.php.net/manual/en/function.exif-imagetype.php
```

up
down
1
Nikki ¶
7 years ago

```
This should be easy, but I've re-solved the problem so many times.  Hopefully this is useful:

<?php
// Usage example to find the proper dimensions to resize an image down to 300x400 pixels maximum:
list($width, $height) = getimagesize($image);
$new_dimensions = resize_dimensions(300,400,$width,$height);

// Calculates restricted dimensions with a maximum of $goal_width by $goal_height
function resize_dimensions($goal_width,$goal_height,$width,$height) {
    $return = array('width' => $width, 'height' => $height);

    // If the ratio > goal ratio and the width > goal width resize down to goal width
    if ($width/$height > $goal_width/$goal_height && $width > $goal_width) {
        $return['width'] = $goal_width;
        $return['height'] = $goal_width/$width * $height;
    }
    // Otherwise, if the height > goal, resize down to goal height
    else if ($height > $goal_height) {
        $return['width'] = $goal_height/$height * $width;
        $return['height'] = $goal_height;
    }

    return $return;
}
?>
```

up
down
1
shmohel at gmail dot com ¶
9 years ago

```
Rather than making a lengthy function that essentially runs twice (once as width, once as height) I came up with a helpful function that uses variable variables to
set a maximum height/width. Hope someone finds this helpful.

function scaleimage($location, $maxw=NULL, $maxh=NULL){
    $img = @getimagesize($location);
    if($img){
        $w = $img[0];
        $h = $img[1];

        $dim = array('w','h');
        foreach($dim AS $val){
            $max = "max{$val}";
            if(${$val} > ${$max} && ${$max}){
                $alt = ($val == 'w') ? 'h' : 'w';
                $ratio = ${$alt} / ${$val};
                ${$val} = ${$max};
                ${$alt} = ${$val} * $ratio;
            }
        }

        return("<img src='{$location}' alt='image' width='{$w}' height='{$h}' />");
    }
}
```

up
down
2
diablx at hotmail dot com ¶
13 years ago

```
I'm sorry for they other scripts, but I made one mistake about the image resizing... here is a working script !
<?
    // Some configuration variables !
    $maxWidth = 90;
    $maxHeight = 90;
    $maxCols = 8;
    $webDir = "https://localhost/images/";
    $localDir = $_SERVER['DOCUMENT_ROOT']."/images/";

    $AutorisedImageType = array ("jpg", "jpeg", "gif", "png");
?>

<center>
<table border='1' cellspacing='5' cellpadding='5' style="border-collapse:collapse; border-style: dotted">
<tr>
   <?
   // Open localDir
   $dh = opendir($localDir);
   while (false !== ($filename = readdir($dh))) {
       $filesArray[] = $filename;
   }
```

```
    // Display and resize
    foreach ($filesArray as $images) {

        $ext = substr($images, strpos($images, ".")+1, strlen($images));

        if( in_array($ext, $AutorisedImageType) ) {

            list($width, $height, $type, $attr) = @getimagesize( $localDir.$images );

             $xRatio = $maxWidth / $width;
             $yRatio = $maxHeight / $height;

             if ( ($width <= $maxWidth) && ($height <= $maxHeight) ) {
               $newWidth = $width;
               $newHeight = $height;
             }
             else if (($xRatio * $height) < $maxHeight) {
               $newHeight = ceil($xRatio * $height);
               $newWidth = $maxWidth;
             }
             else {
               $newWidth = ceil($yRatio * $width);
               $newHeight = $maxHeight;
             }

            if($i == $maxCols) {
                echo "</tr><tr>";
                $i = 0;
            }
            echo "<td align='center' valign='middle' width='$maxWidth' height='$maxHeight'><img src='".$webDir.$images."' width='$newWidth' height='$newHeight'></td>";
            $i++;
        }
    }
?>
</tr>
</table>
</center>
```

up
down
1
redcore at gmail dot com ¶
10 years ago

```
It's always good to check out an image's dimensions while attempting to upload to your server or database...especially if it's going to be displayed on a page that
doesn't accomodate images beyond a particular size.

<?php

$tmpName = $_FILES['userfile']['tmp_name'];

list($width, $height, $type, $attr) = getimagesize($tmpName);

if($width>275 || $height>275)
{
die("exceeded image dimension limits.");
}
```

```
?>
```
up
down
1
laurens dot stoetzel at gmail dot com ¶
10 years ago
```
In reply to John (http://de.php.net/manual/de/function.getimagesize.php#61514):
list will only work with numeric arrays.

<?php
  //renumber
  $my_image = array_values(getimagesize('test.jpg'));
  //use list on new array
  list($width, $height, $type, $attr) = $my_image;

  //view new array
  print_r($my_image);

  //spit out content
  echo 'Attribute: '.$attr.'<br />';
  echo 'Width: '.$width.'<br />';
?>
```
up
down
1
utilmind ¶
6 years ago
```
Here is the function which determines whether the PNG image contains alpha or not:

<?php
function is_alpha_png($fn){
  return (ord(@file_get_contents($fn, NULL, NULL, 25, 1)) == 6);
}
?>

The color type of PNG image is stored at byte offset 25. Possible values of that 25'th byte is:
* 0 - greyscale
* 2 - RGB
* 3 - RGB with palette
* 4 - greyscale + alpha
* 6 - RGB + alpha
```
up
down
0
freecorvette at gmail dot com ¶
8 days ago
For some images, using getimagesize() without the second parameter will return the correct info, but when you add the second parameter it will return false. This is most likely a bug (and it has been reported as such), but meanwhile, if you encounter this problem, a workaround is to use exif_read_data().
up
down
0
charliehu dot cn at hotmail dot com ¶
1 month ago

```
If call getimagesize with a nonexistent url then it would be entering a dead loop, and the access_log of apache will print a lot of access url log continuously.

It would be sure to occur when call getimagesize with a nonexistent url. I don't know why.

The access_log intercept ( the url is "/wordpress/wp-content/uploads/2016/08/logo.png")

::1 - - [24/Aug/2017:10:03:33 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:33 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:34 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:34 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:34 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:35 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:35 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:36 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:36 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:37 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:37 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:38 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:38 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:38 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
::1 - - [24/Aug/2017:10:03:39 +0800] "GET /wordpress/wp-content/uploads/2016/08/logo.png HTTP/1.0" 404 20078
```

up
down
0
chris at ocportal dot com ¶
5 years ago

```
Note that animated gifs may have frames width different dimensions. This function will not get the first frame's width/height. GIFs define "Logical Screen Descriptor" dimensions, which are the maximum for all frames.

To get the screen descriptor dimensions, use:

<?php
$header = unpack('@6/vwidth/vheight', $binaryData);
// $header['width'] and $header['width'];
?>
```

up
down
0
user at example dot net ¶
9 years ago

```
When validating images, allways check both, image type *AND* file extension!

Because most image types allow sections for comments or other irrelevant data. Those section can be used to infiltrate php code onto the server. If these files are stored as sent by the client, files with a ".php" extension can be executed and do tremendous harm.
```

up
down
0
cstdenis at hotmail dot com ¶
13 years ago

```
This will not work for swf files unless zlib is compiled into php statically (not as a shared module). Bug #29611

As of PHP 5.0.0 it will just return false, but that should change to a notice by the next release.
```

up
down
-1

paul at goldenbakery dot nl ¶
12 years ago
```
Note that the canvas of a Flash movie can not be empty for getimagesize() to read the dimensions of an SWF. Not sure if this is a bug, a feature or just a limitation of the SWF format.

Flash version does not seem to matter. Also tested with Flash 8 beta.
```
up
down
-6
anonymous ¶
8 years ago
```
Note that if you specify a remote file (via a URL) to check the size of, PHP will first download the remote file to your server.

If you're using this function to check the size of user provided image links, this could constitute a security risk.  A malicious user could potentially link to a very large image file and cause PHP to download it.  I do not know what, if any, file size limits are in place for the download.  But suppose the user provided a link to an image that was several gigabytes in size?

It would be nice if there were a way to limit the size of the download performed by this function.  Hopefully there is already a default with some sensible limits.
```
add a note