Exchangeable image information

Introduction

With the exif extension you are able to work with image meta data. For example, you may use exif functions to read meta data of pictures taken from digital cameras by working with information stored in the headers of the JPEG and TIFF images.

Installing/Configuring

Requirements

Your PHP must be compiled in with *--enable-exif*. PHP does not require any additional library for the exif module. Windows users must also have the mbstring extension enabled.

Installation

To enable exif-support configure PHP with *--enable-exif*

Windows users must enable both the *php_mbstring.dll* and *php_exif.dll* DLL's in *php.ini*. The *php_mbstring.dll* DLL must be loaded *before* the *php_exif.dll* DLL so adjust your *php.ini* accordingly.

Runtime Configuration

The behaviour of these functions is affected by settings in *php.ini*.

Exif supports automatically conversion for Unicode and JIS character encodings of user comments when module mbstring is available. This is done by first decoding the comment using the specified characterset. The result is then encoded with another characterset which should match your *HTTP* output.

Exif configuration options

Name	Default	Changeable	Changelog
exif.encode_unicode	"ISO-8859-15"	PHP_INI_ALL	Available since PHP 4.3.0.
exif.decode_unicode_motorola	"UCS-2BE"	PHP_INI_ALL	Available since PHP 4.3.0.
exif.decode_unicode_intel	"UCS-2LE"	PHP_INI_ALL	Available since PHP 4.3.0.
exif.encode_jis	""	PHP_INI_ALL	Available since PHP 4.3.0.
exif.decode_jis_motorola	"JIS"	PHP_INI_ALL	Available since PHP 4.3.0.
exif.decode_jis_intel	"JIS"	PHP_INI_ALL	Available since PHP 4.3.0.

For further details and definitions of the PHP_INI_* constants, see the php.ini directives.

Here's a short explanation of the configuration directives.

`exif.encode_unicode` string
: *exif.encode_unicode* defines the characterset UNICODE user comments are handled. This defaults to ISO-8859-15 which should work for most non Asian countries. The setting can be empty or must be an encoding supported by mbstring. If it is empty the current internal encoding of mbstring is used.

`exif.decode_unicode_motorola` string
: *exif.decode_unicode_motorola* defines the image internal characterset for Unicode encoded user comments if image is in motorola byte order (big-endian). This setting cannot be empty but you can specify a list of encodings supported by mbstring. The default is UCS-2BE.

`exif.decode_unicode_intel` string
: *exif.decode_unicode_intel* defines the image internal characterset for Unicode encoded user comments if image is in intel byte order (little-endian). This setting cannot be empty but you can specify a list of encodings supported by mbstring. The default is UCS-2LE.

`exif.encode_jis` string
: *exif.encode_jis* defines the characterset JIS user comments are handled. This defaults to an empty value which forces the functions to use the current internal encoding of mbstring.

`exif.decode_jis_motorola` string
: *exif.decode_jis_motorola* defines the image internal characterset for JIS encoded user comments if image is in motorola byte order (big-endian). This setting cannot be empty but you can specify a list of encodings supported by mbstring. The default is JIS.

`exif.decode_jis_intel` string
: *exif.decode_jis_intel* defines the image internal characterset for JIS encoded user comments if image is in intel byte order (little-endian). This setting cannot be empty but you can specify a list of encodings supported by mbstring. The default is JIS.

Resource Types

This extension has no resource types defined.

Predefined Constants

The constants below are defined by this extension, and will only be available when the extension has either been compiled into PHP or dynamically loaded at runtime.

EXIF_USE_MBSTRING (integer)

The exif_imagetype() lists several related built-in constants.

Exif Functions

exif_imagetype

exif_imagetype -- Determine the type of an image

Description

int **exif_imagetype** (string `$filename`)

exif_imagetype() reads the first bytes of an image and checks its signature.

exif_imagetype() can be used to avoid calls to other exif functions with unsupported file types or in conjunction with *$_SERVER['HTTP_ACCEPT']* to check whether or not the viewer is able to see a specific image in the browser.

Parameters

`filename`
The image being checked.

Return Values

When a correct signature is found, the appropriate constant value will be returned otherwise the return value is **FALSE**. The return value is the same value that getimagesize() returns in index 2 but exif_imagetype() is much faster.

ChangeLog

Version	Description
4.3.2	Support for JPC, JP2, JPX, JB2, XBM, and WBMP
4.3.0	Support for SWC

Predefined Constants

The following constants are defined, and represent possible exif_imagetype() return values:

Imagetype Constants

Value	Constant

1	**IMAGETYPE_GIF**
2	**IMAGETYPE_JPEG**
3	**IMAGETYPE_PNG**
4	**IMAGETYPE_SWF**
5	**IMAGETYPE_PSD**
6	**IMAGETYPE_BMP**
7	**IMAGETYPE_TIFF_II** (intel byte order)
8	**IMAGETYPE_TIFF_MM** (motorola byte order)
9	**IMAGETYPE_JPC**
10	**IMAGETYPE_JP2**
11	**IMAGETYPE_JPX**
12	**IMAGETYPE_JB2**
13	**IMAGETYPE_SWC**
14	**IMAGETYPE_IFF**
15	**IMAGETYPE_WBMP**
16	**IMAGETYPE_XBM**

Examples

Example #1 - exif_imagetype() example

```
<?php
if (exif_imagetype('image.gif') != IMAGETYPE_GIF) {
   echo 'The picture is not a gif';
}
?>
```

See Also

- getimagesize()

exif_read_data

exif_read_data -- Reads the EXIF headers from JPEG or TIFF

Description

array **exif_read_data** (string `$filename` [, string `$sections` [, bool `$arrays` [, bool `$thumbnail`]]])

exif_read_data() reads the EXIF headers from a JPEG or TIFF image file. This way you can read meta data generated by digital cameras.

Exif headers tend to be present in JPEG/TIFF images generated by digital cameras, but unfortunately each digital camera maker has a different idea of how to actually tag their images, so you can't always rely on a specific Exif header being present.

Height and *Width* are computed the same way getimagesize() does so their values must not be part of any header returned. Also, *html* is a height/width text string to be used inside normal HTML.

When an Exif header contains a Copyright note, this itself can contain two values. As the solution is inconsistent in the Exif 2.10 standard, the COMPUTED section will return both entries *Copyright.Photographer* and *Copyright.Editor* while the IFD0 sections contains the byte array with the NULL character that splits both entries. Or just the first entry if the datatype was wrong (normal behaviour of Exif). The COMPUTED will also contain the entry *Copyright* which is either the original copyright string, or a comma separated list of the photo and editor copyright.

The tag UserComment has the same problem as the Copyright tag. It can store two values. First the encoding used, and second the value itself. If so the IFD section only contains the encoding or a byte array. The COMPUTED section will store both in the entries *UserCommentEncoding* and *UserComment*. The entry *UserComment* is available in both cases so it should be used in preference to the value in IFD0 section.

exif_read_data() also validates EXIF data tags according to the EXIF specification (» http://exif.org/Exif2-2.PDF, page 20).

Note
Windows ME/XP can both wipe the Exif headers when connecting to a camera. More information available at » http://www.canon.co.jp/Imaging/NOTICE/011214-e.html.

Parameters

`filename`
The name of the image file being read. This cannot be an URL.

sections
Is a comma separated list of sections that need to be present in file to produce a result array. If none of the requested sections could be found the return value is **FALSE**.

FILE	FileName, FileSize, FileDateTime, SectionsFound
COMPUTED	html, Width, Height, IsColor, and more if available. Height and Width are computed the same way getimagesize() does so their values must not be part of any header returned. Also, html is a height/width text string to be used inside normal HTML.
ANY_TAG	Any information that has a Tag e.g. IFD0, EXIF, ...
IFD0	All tagged data of IFD0. In normal imagefiles this contains image size and so forth.
THUMBNAIL	A file is supposed to contain a thumbnail if it has a second IFD. All tagged information about the embedded thumbnail is stored in this section.
COMMENT	Comment headers of JPEG images.
EXIF	The EXIF section is a sub section of IFD0. It contains more detailed information about an image. Most of these entries are digital camera related.

arrays
Specifies whether or not each section becomes an array. The *sections* *COMPUTED*, *THUMBNAIL*, and *COMMENT* always become arrays as they may contain values whose names conflict with other sections.

thumbnail
When set to **TRUE** the thumbnail itself is read. Otherwise, only the tagged data is read.

Return Values

It returns an associative array where the array indexes are the header names and the array values are the values associated with those headers. If no data can be returned, exif_read_data() will return **FALSE**.

ChangeLog

Version	Description
4.3.0	Can read all embedded IFD data including arrays (returned as such). Also the size of an embedded thumbnail is returned in a *THUMBNAIL* subarray, and can return thumbnails in TIFF format. Also, there is no longer a maximum length for returned values (not until the memory limit has been reached)
4.3.0	If PHP has mbstring support, the user comment can automatically change encoding. Also, if the user comment uses Unicode or JIS encoding this encoding will automatically be changed according to the exif ini settings in *php.ini*
4.3.0	If the image contains any IFD0 data then COMPUTED contains the entry ByteOrderMotorola which is 0 for little-endian (intel) and 1 for big-endian (motorola) byte order. Also, COMPUTED and UserComment no longer only contain the first copyright entry if the datatype was wrong.

Examples

Example #2 - exif_read_data() example

```
<?php
echo "test1.jpg:<br />\n";
$exif = exif_read_data('tests/test1.jpg', 'IFD0');
echo $exif===false ? "No header data found.<br />\n" : "Image contains
headers<br />\n";

$exif = exif_read_data('tests/test2.jpg', 0, true);
echo "test2.jpg:<br />\n";
foreach ($exif as $key => $section) {
   foreach ($section as $name => $val) {
       echo "$key.$name: $val<br />\n";
   }
}
?>
```

The first call fails because the image has no header information.

The above example will output something similar to:

```
test1.jpg:
No header data found.
test2.jpg:
FILE.FileName: test2.jpg
FILE.FileDateTime: 1017666176
FILE.FileSize: 1240
FILE.FileType: 2
FILE.SectionsFound: ANY_TAG, IFD0, THUMBNAIL, COMMENT
COMPUTED.html: width="1" height="1"
COMPUTED.Height: 1
COMPUTED.Width: 1
COMPUTED.IsColor: 1
COMPUTED.ByteOrderMotorola: 1
COMPUTED.UserComment: Exif test image.
COMPUTED.UserCommentEncoding: ASCII
COMPUTED.Copyright: Photo (c) M.Boerger, Edited by M.Boerger.
COMPUTED.Copyright.Photographer: Photo (c) M.Boerger
COMPUTED.Copyright.Editor: Edited by M.Boerger.
IFD0.Copyright: Photo (c) M.Boerger
IFD0.UserComment: ASCII
THUMBNAIL.JPEGInterchangeFormat: 134
THUMBNAIL.JPEGInterchangeFormatLength: 523
COMMENT.0: Comment #1.
COMMENT.1: Comment #2.
COMMENT.2: Comment #3end
THUMBNAIL.JPEGInterchangeFormat: 134
THUMBNAIL.Thumbnail.Height: 1
THUMBNAIL.Thumbnail.Height: 1
```

See Also

- exif_thumbnail()
- getimagesize()

exif_tagname

exif_tagname -- Get the header name for an index

Description

string **exif_tagname** (string `$index`)

Warning
This function is currently not documented; only its argument list is available.

Parameters

`index`
The image index

Return Values

Returns the header name, or **FALSE** if *`index`* is undefined.

See Also

- exif_imagetype()

exif_thumbnail

exif_thumbnail -- Retrieve the embedded thumbnail of a TIFF or JPEG image

Description

string **exif_thumbnail** (string `$filename` [, int `&$width` [, int `&$height` [, int `&$imagetype`]]])

exif_thumbnail() reads the embedded thumbnail of a TIFF or JPEG image.

If you want to deliver thumbnails through this function, you should send the mimetype information using the header() function.

It is possible that exif_thumbnail() cannot create an image but can determine its size. In this case, the return value is **FALSE** but *`width`* and *`height`* are set.

Parameters

`filename`
The name of the image file being read. This image contains an embedded thumbnail.

`width`
The return width of the returned thumbnail.

`height`
The returned height of the returned thumbnail.

`imagetype`
The returned image type of the returned thumbnail. This is either TIFF or JPEG.

Return Values

Returns the embedded thumbnail, or **FALSE** if the image contains no thumbnail.

ChangeLog

Version	Description
4.3.0	The optional parameters *`width`*, *`height`*, and *`imagetype`* all became available.
4.3.0	May return thumbnails in the TIFF format.

Examples

Example #3 - exif_thumbnail() example

```
<?php
if (array_key_exists('file', $_REQUEST)) {
   $image = exif_thumbnail($_REQUEST['file'], $width, $height, $type);
} else {
   $image = false;
}
if ($image!==false) {
   header('Content-type: ' .image_type_to_mime_type($type));
   echo $image;
   exit;
} else {
   // no thumbnail available, handle the error here
   echo 'No thumbnail available';
}
?>
```

See Also

- exif_read_data()
- image_type_to_mime_type()

read_exif_data

read_exif_data -- Alias of exif_read_data()

Description

This function is an alias of: exif_read_data().