

---

# **pytoshop Documentation**

*Release 1.2.1*

**Michael Droettboom**

**Nov 30, 2018**



---

# Contents

---

<b>1</b>	<b>pytoshop</b>	<b>3</b>
1.1	Features . . . . .	3
<b>2</b>	<b>Installation</b>	<b>5</b>
2.1	Stable release . . . . .	5
2.2	From sources . . . . .	5
<b>3</b>	<b>Usage</b>	<b>7</b>
<b>4</b>	<b>API</b>	<b>9</b>
4.1	User api . . . . .	9
4.2	Objects . . . . .	11
4.3	Enums . . . . .	30
4.4	Codecs . . . . .	38
4.5	Utilities . . . . .	42
<b>5</b>	<b>Credits</b>	<b>47</b>
5.1	Development Lead . . . . .	47
5.2	Contributors . . . . .	47
<b>6</b>	<b>History</b>	<b>49</b>
6.1	1.2.1 (2018-11-30) (2017-12-13) (2017-09-26) . . . . .	49
6.2	1.0.1 (2017-08-01) (2017-07-29) . . . . .	49
6.3	0.6.0 (2017-07-24) . . . . .	49
6.4	0.5.0 (2017-07-03) . . . . .	49
6.5	0.4.1 (2017-06-05) . . . . .	50
6.6	0.4.0 (2017-05-12) . . . . .	50
6.7	0.3.0 (2017-01-09) . . . . .	50
	<b>Python Module Index</b>	<b>51</b>



Python library for reading and writing complex structured Photoshop files.

Contents:



A Python-based library to read and write Photoshop PSD and PSB files.

Based on the specification from [Adobe](#), but also with the help of the [psd-tools](#) source code.

- Free software: BSD license
- Documentation: <https://pytoshop.readthedocs.io>.

## 1.1 Features

- Parsing of the most important tags. This is not complete, but the infrastructure is in place to add support for more quite easily.
- Loading of complex nested layer structures, and the ability to edit them and write them back out.





### 2.1 Stable release

To install pytoshop, run this command in your terminal:

```
$ pip install pytoshop
```

This is the preferred method to install pytoshop, as it will always install the most recent stable release.

If you don't have `pip` installed, this [Python installation guide](#) can guide you through the process.

### 2.2 From sources

The sources for pytoshop can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/mdboom/pytoshop
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/mdboom/pytoshop/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```



To read a file and write it back out again:

```
import pytoshop

with open('image.psd', 'rb') as fd:
    psd = pytoshop.read(fd)

    with open('updated.psd', 'wb') as fd:
        psd.write(fd)
```

See the [API](#) documentation for more details.



---

## 4.1 User api

<code>read(fd)</code>	Read a PSD file from a file-like object.
<code>user.nested_layers</code>	Convert a PSD file to/from nested layers.

### 4.1.1 pytoshop.read

`pytoshop.read(fd)`

Read a PSD file from a file-like object.

**Parameters** `fd` (*file-like object*) – Must be readable, seekable and open in binary mode.

**Returns** `psdfile`

**Return type** `PsdFile`

### 4.1.2 pytoshop.user.nested\_layers

Convert a PSD file to/from nested layers.

```
class pytoshop.user.nested_layers.Group (name="", visible=True, opacity=255, group_id=0,
                                         blend_mode=<BlendMode.pass_through:
                                         b'pass'>, layers=None, closed=True, meta-
                                         data=None, layer_color=0)
```

Bases: `pytoshop.user.nested_layers.Layer`

A `Layer` that may contain other `Layer` instances.

**closed**

Is layer closed in GUI?

**layers**

List of sublayers

```
class pytoshop.user.nested_layers.Image (name="", visible=True, opacity=255, group_id=0,
                                          blend_mode=<BlendMode.normal: b'norm'>,
                                          top=0, left=0, bottom=None, right=None, chan-
                                          nels=None, metadata=None, layer_color=0,
                                          color_mode=None)
```

Bases: `pytoshop.user.nested_layers.Layer`

A `Layer` containing image data, i.e. a leaf node.

**bottom**

The bottom of the layer, in pixels. If not provided, will be automatically determined from channel data.

**channels**

The channel image data. May be one of the following –

- A dictionary from `enums.ChannelId` to 2-D numpy arrays.
- A 3-D numpy array of the shape (num\_channels, height, width)
- A list of numpy arrays where each is a channel.

It is better to use `get_channel` and `set_channel` to

**color\_mode**

The color mode of the image.

**get\_channel** (*color*)

Get a channel for a given color. Raises an error if the color space doesn't have the given color.

**Parameters** **color** (`enums.ColorChannel`) –

**Returns** **channel**

**Return type** 2-D numpy array

**left**

The left of the layer, in pixels.

**right**

The right of the layer, in pixels. If not provided, will be automatically determined from channel data.

**set\_channel** (*color*, *channel*)

Get a channel for a given color. Raises an error if the color space doesn't have the given color.

**Parameters**

- **color** (`enums.ColorChannel`) –
- **channel** (2-D numpy array) –

**top**

The top of the layer, in pixels.

```
class pytoshop.user.nested_layers.Layer
```

Bases: `object`

Base class of all layers.

**blend\_mode**

blend mode

**group\_id**

Linked layer id

**layer\_color**

layer color (as it appears in the layer list)

**metadata****name**

The name of the layer

**opacity**

Opacity. 0=transparent, 255=opaque

**visible**

Is layer visible?

```
pytoshop.user.nested_layers.nested_layers_to_psd(layers, color_mode, version=<Version.version_1: 1>,
                                                  compression=<Compression.rle: 1>, depth=None, size=None, vector_mask=False)
```

Convert a hierarchy of nested *Layer* instances to a *PsdFile*.

**Parameters**

- **layers** (list of *Layer* instances) – The hierarchy of layers we want to create.
- **color\_mode** (*enums.ColorMode*) – The color mode of the resulting PSD file (as well as the input image data).
- **version** (*enums.Version*, optional) – The version of the PSD spec to follow.
- **compression** (*enums.Compression*, optional) – The method of image compression to use for the layer image data.
- **depth** (*enums.ColorDepth*, optional) – The color depth of the resulting image. Must match the color depth of the data passed in. If not provided, the color depth will be automatically determined from the passed-in data.
- **size** (2-tuple of *int*, optional) – The shape in the form (height, width) of the resulting PSD file. If not provided, the height and width will be set to include all passed in layers, and the layers themselves will be adjusted so that none fall outside of the image.
- **vector\_mask** (*bool*, optional) – When `True`, the mask for the layer will be a vector rectangle. This results in much smaller file sizes, but is not quite as accurately rendered by Photoshop. When `False`, a raster mask is used.

**Returns** *psdfile* – The resulting PSD file.

**Return type** *PsdFile*

```
pytoshop.user.nested_layers.pprint_layers(layers, indent=0)
```

Pretty-print a hierarchy of *Layer* instances.

```
pytoshop.user.nested_layers.psd_to_nested_layers(psdfile)
```

Convert a *PsdFile* instance to a hierarchy of nested *Layer* instances.

**Parameters** *psdfile* (*PsdFile*) – A parsed PSD file.

**Returns** *layers* – A representation of the parsed hierarchy from the file.

**Return type** list of *Layer* instances

## 4.2 Objects

<code>blending_range</code>	Manage blending ranges.
<code>core</code>	The core objects, including the <code>PsdFile</code> and its <code>Header</code> .
<code>color_mode</code>	The <code>ColorModeData</code> section.
<code>image_data</code>	The <code>ImageData</code> section.
<code>image_resources</code>	The <code>ImageResources</code> section.
<code>layers</code>	Sections related to image layers.
<code>path</code>	Handle Bézier paths.
<code>tagged_block</code>	<code>TaggedBlock</code> objects.

## 4.2.1 pytoshop.blending\_range

Manage blending ranges.

**class** `pytoshop.blending_range.BlendingRange` (*black0=0, black1=0, white0=0, white1=0, \_values=None*)

Bases: `object`

Blending range data.

Comprises 2 black values and 2 white values.

**black0**

**black1**

**classmethod** `read` (*fd*)

Instantiate from a file-like object.

### Parameters

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**white0**

**white1**

**write** (*fd, header*)

Write to a file-like object.

### Parameters

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**class** `pytoshop.blending_range.BlendingRangePair` (*src=None, dst=None*)

Bases: `object`

Blending range pair.

The combination of a source and destination blending range.

**dst**

**length** (*header*)

The length of the section, in bytes, not including its header.

**classmethod** `read` (*fd*)

Instantiate from a file-like object.

### Parameters



- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**src**

**total\_length** (*header*)

The length of the section, in bytes, including its header.

**write** (*fd, header*)

Write to a file-like object.

#### Parameters

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**class** `pytoshop.blending_range.BlendingRanges` (*composite\_gray\_blend=None, channels=None*)

Bases: `object`

All of the layer blending range data.

Consists of a composite gray blend pair followed by N additional pairs.

**channels**

List of additional *BlendingRangePair* instances.

**composite\_gray\_blend**

Composite gray *BlendingRangePair*.

**length** (*header*)

The length of the section, in bytes, not including its header.

**classmethod read** (*fd, num\_channels*)

Instantiate from a file-like object.

#### Parameters

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**total\_length** (*header*)

The length of the section, in bytes, including its header.

**write** (*fd, header*)

Write to a file-like object.

#### Parameters

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

## 4.2.2 pytoshop.core

The core objects, including the *PsdFile* and its *Header*.

**class** `pytoshop.core.Header` (*version=<Version.version\_1: 1>, num\_channels=1, height=1, width=1, depth=<ColorDepth.depth8: 8>, color\_mode=<ColorMode.rgb: 3>*)

Bases: `object`

Manages the header at the start of a PSD/PSB file.

**color\_mode**

Color mode of the file. See *enums.ColorMode*.

**depth**

Number of bits per channel. See *enums.ColorDepth*.

**classmethod header\_read** (*fd*)

Instantiate from a file-like object.

**Parameters** *fd* (*file-like object*) – Must be readable, seekable and open in binary mode.

**height**

Height of the image (in pixels).

**max\_size\_mapping** = {1: 30000, 2: 300000}

**num\_channels**

Number of color channels in the file.

**shape**

**version**

The version of the file format. See *enums.Version*.

**width**

Width of the image (in pixels).

**write** (*fd*)

Write to a file-like object.

**Parameters** *fd* (*file-like object*) – Must be writable, seekable and open in binary mode.

```
class pytoshop.core.PsdFile (version=<Version.version_1: 1>, num_channels=1,
                             height=1, width=1, depth=<ColorDepth.depth8: 8>,
                             color_mode=<ColorMode.rgb: 3>, color_mode_data=None,
                             image_resources=None, layer_and_mask_info=None,
                             image_data=None, compression=<Compression.raw: 0>)
```

Bases: *pytoshop.core.Header*

Represents an entire PSD file.

**color\_mode\_data**

Color mode data section. See *color\_mode.ColorModeData*.

**image\_data**

Image data. See *image\_data.ImageData*.

**image\_resources**

Image resources. See *image\_resources.ImageResources*.

**layer\_and\_mask\_info**

Image resources. See *image\_resources.ImageResources*.

**classmethod read** (*fd*)

Instantiate from a file-like object.

**Parameters** *fd* (*file-like object*) – Must be readable, seekable and open in binary mode.

**write** (*fd*)

Write to a file-like object.

**Parameters** `fd` (*file-like object*) – Must be writable, seekable and open in binary mode.

### 4.2.3 pytoshop.color\_mode

The *ColorModeData* section.

```
class pytoshop.color_mode.ColorModeData (data=b")
    Bases: object
```

Color mode data section.

Only indexed color and duotone (see *core.Header.color\_mode*) have color mode data.

Indexed color images: length is 768; color data contains the color table for the image, in non-interleaved order.

Duotone images: color data contains the duotone specification (the format of which is not documented). Other applications that read Photoshop files can treat a duotone image as a gray image, and just preserve the contents of the duotone information when reading and writing the file.

Note that `pytoshop` doesn't do anything meaningful for color mode data, and only stores the raw bytes in order to round-trip.

**data**

**length** (*header*)

The length of the section, in bytes, not including its header.

**classmethod read** (*fd, header*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**write** (*fd, header*)

Write to a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

### 4.2.4 pytoshop.image\_data

The *ImageData* section.

```
class pytoshop.image_data.ImageData (channels=None, fd=None, offset=None, size=None,
                                     height=None, width=None, num_channels=None,
                                     depth=None, version=None, compression=<Compression.raw: 0>)
```

Bases: `object`

Stores (non-layer) image data.

**channels**

**compression**

Type of compression. See *enums.Compression*.

```

classmethod read (fd, header)
shape
write (fd, header)

```

## 4.2.5 pytoshop.image\_resources

The *ImageResources* section.

Image resource blocks are the basic building unit of several file formats, including Photoshop's native file format, JPEG, and TIFF. Image resources are used to store non-pixel data associated with images, such as pen tool paths.

```

class pytoshop.image_resources.AlphaIdentifiers (name="", identifiers=[])

```

Bases: *pytoshop.image\_resources.ImageResourceBlock*

```

data_length (header)

```

```

identifiers

```

Alpha indentifiers

```

classmethod read_data (fd, resource_id, name, length, header)

```

```

write_data (fd, header)

```

```

class pytoshop.image_resources.BackgroundColor (name="",

```

```

color_space=<ColorSpace.rgb: 0>,

```

```

color=[])

```

Bases: *pytoshop.image\_resources.ImageResourceBlock*

Background color.

```

color

```

The color data. If the color data does not require 4 values, the extra values are undefined and should be included as zeros.

```

color_space

```

The color space. See *enums.ColorSpace*

```

data_length (header)

```

```

classmethod read_data (fd, resource_id, name, length, header)

```

```

write_data (fd, header)

```

```

class pytoshop.image_resources.BorderInfo (name="", border_width_num=0, bor-

```

```

der_width_den=1, unit=<Units.inches: 1>)

```

Bases: *pytoshop.image\_resources.ImageResourceBlock*

Border information.

```

border_width_den

```

Border width denominator

```

border_width_num

```

Border width numerator

```

data_length (header)

```

```

classmethod read_data (fd, resource_id, name, length, header)

```

```

unit

```

Unit. See *enums.Units*.

```

write_data (fd, header)

```

```

class pytoshop.image_resources.CopyrightFlag (name="", copyright=False)
    Bases: pytoshop.image_resources.ImageResourceBlock

    copyright
        Is copyrighted?

    data_length (header)

    classmethod read_data (fd, resource_id, name, length, header)

    write_data (fd, header)

class pytoshop.image_resources.DocumentSpecificIdsSeedNumber (name="",
                                                                base_value=0)
    Bases: pytoshop.image_resources.ImageResourceBlock

    base_value
        Base value

    data_length (header)

    classmethod read_data (fd, resource_id, name, length, header)

    write_data (fd, header)

class pytoshop.image_resources.EffectsVisible (name="", visible=False)
    Bases: pytoshop.image_resources.ImageResourceBlock

    data_length (header)

    classmethod read_data (fd, resource_id, name, length, header)

    visible
        Are effects visible?

    write_data (fd, header)

class pytoshop.image_resources.GenericImageResourceBlock (name="", resource_id=0,
                                                            data=b"")
    Bases: pytoshop.image_resources.ImageResourceBlock

    data
        Raw data of image resource.

    data_length (header)

    classmethod read_data (fd, resource_id, name, length, header)

    resource_id
        Type of image resource.

    write_data (fd, header)

class pytoshop.image_resources.GlobalAltitude (name="", altitude=0)
    Bases: pytoshop.image_resources.ImageResourceBlock

    altitude
        Global altitude

    data_length (header)

    classmethod read_data (fd, resource_id, name, length, header)

    write_data (fd, header)

class pytoshop.image_resources.GlobalAngle (name="", angle=0)
    Bases: pytoshop.image_resources.ImageResourceBlock

```

**angle**

Global light angle for the effect layer

**data\_length** (*header*)

**classmethod read\_data** (*fd, resource\_id, name, length, header*)

**write\_data** (*fd, header*)

**class** `pytoshop.image_resources.GridAndGuidesInfo` (*name=""*, *grid\_hori=0*, *grid\_vert=0*,  
*guides=[]*)

Bases: `pytoshop.image_resources.ImageResourceBlock`

Grid and guides resource.

**data\_length** (*header*)

**grid\_hori**

Document-specific grid (horizontal). In 1/32 pt.

**grid\_vert**

Document-specific grid (vertical). In 1/32 pt.

**guides**

Guides. See `GuideResourceBlock`.

**classmethod read\_data** (*fd, resource\_id, name, length, header*)

**version**

**write\_data** (*fd, header*)

**class** `pytoshop.image_resources.GuideResourceBlock` (*location=0*, *direction=<GuideDirection.vertical: 0>*)

Bases: `object`

**data\_length** (*header*)

**direction**

Guide direction. See `enums.GuideDirection`.

**location**

Location of guide in document coordinates.

**classmethod read** (*fd, header*)

**write** (*fd, header*)

**class** `pytoshop.image_resources.ImageResourceBlock`

Bases: `object`

Stores a single image resource block.

`pytoshop` currently doesn't deeply parse image resource blocks. The raw data is merely retained for round-tripping.

**data\_length** (*header*)

**length** (*header*)

The length of the section, in bytes, not including its header.

**name**

Name of image resource.

**classmethod read** (*fd, header*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**classmethod read\_data** (*fd, resource\_id, name, data\_length, header*)

**resource\_id**

Type of image resource.

**total\_length** (*header*)

The length of the section, in bytes, including its header.

**write** (*fd, header*)

Write to a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**write\_data** (*fd, header*)

**class** `pytoshop.image_resources.ImageResourceUnicodeString` (*name="", value=""*)

Bases: `pytoshop.image_resources.ImageResourceBlock`

**data\_length** (*header*)

**classmethod read\_data** (*fd, resource\_id, name, length, header*)

**value**

**write\_data** (*fd, header*)

**class** `pytoshop.image_resources.ImageResources` (*blocks=[]*)

Bases: `object`

The image resource block section.

**blocks**

List of all `ImageResourceBlock` items.

**get\_block** (*resource\_id*)

Get the first block with the given resource id.

**length** (*header*)

The length of the section, in bytes, not including its header.

**classmethod read** (*fd, header*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**total\_length** (*header*)

The length of the section, in bytes, including its header.

**write** (*fd, header*)

Write to a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.

- **header** (*PsdFile* object) – An object to get global file information from.

```
class pytoshop.image_resources.LayersGroupInfo (name="", group_ids=[])
    Bases: pytoshop.image_resources.ImageResourceBlock
```

Layers group information.

Indicates which layers are locked together.

**data\_length** (*header*)

**group\_ids**

**classmethod read\_data** (*fd, resource\_id, name, length, header*)

**write\_data** (*fd, header*)

```
class pytoshop.image_resources.PrintFlags (name="", labels=False, crop_marks=False,
                                             color_bars=False, registration_marks=False,
                                             negative=False, flip=False, interpolate=False,
                                             caption=False, print_flags=False)
    Bases: pytoshop.image_resources.ImageResourceBlock
```

Print flags.

**caption**

**color\_bars**  
color bars

**crop\_marks**  
crop marks

**data\_length** (*header*)

**flip**

**interpolate**

**labels**

**negative**

**print\_flags**  
print flags

**classmethod read\_data** (*fd, resource\_id, name, length, header*)

**registration\_marks**  
registration marks

**write\_data** (*fd, header*)

```
class pytoshop.image_resources.PrintScale (name="", style=<PrintScaleStyle.centered: 0>,
                                             x=0.0, y=0.0, scale=0.0)
    Bases: pytoshop.image_resources.ImageResourceBlock
```

**data\_length** (*header*)

**classmethod read\_data** (*fd, resource\_id, name, length, header*)

**scale**

**style**  
Style. See *enums.PrintScaleStyle*.

**write\_data** (*fd, header*)



**x**  
x location

**y**  
y location

**class** `pytoshop.image_resources.UnicodeAlphaNames` (*name=""*, *value=""*)

Bases: `pytoshop.image_resources.ImageResourceUnicodeString`

**class** `pytoshop.image_resources.Url` (*name=""*, *url=b""*)

Bases: `pytoshop.image_resources.ImageResourceBlock`

**data\_length** (*header*)

**classmethod** `read_data` (*fd*, *resource\_id*, *name*, *length*, *header*)

**url**  
URL

**write\_data** (*fd*, *header*)

**class** `pytoshop.image_resources.VersionInfo` (*name=""*, *version=0*,  
*has\_real\_merged\_data=False*, *writer=""*,  
*reader=""*, *file\_version=0*)

Bases: `pytoshop.image_resources.ImageResourceBlock`

**data\_length** (*header*)

**file\_version**  
file version

**has\_real\_merged\_data**  
has real merged data?

**classmethod** `read_data` (*fd*, *resource\_id*, *name*, *length*, *header*)

**reader**  
reader name

**version**

**write\_data** (*fd*, *header*)

**writer**  
writer name

**class** `pytoshop.image_resources.WorkflowUrl` (*name=""*, *value=""*)

Bases: `pytoshop.image_resources.ImageResourceUnicodeString`

## 4.2.6 pytoshop.layers

Sections related to image layers.

**class** `pytoshop.layers.ChannelImageData` (*image=None*, *fd=None*, *offset=None*, *size=None*,  
*shape=None*, *depth=None*, *version=None*, *compression=<Compression.raw: 0>*)

Bases: `object`

A single plane of channel image data.

**compression**  
Compression method. See `enums.Compression`.

**dtype**

**image**

**classmethod read** (*fd, header, shape, size*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**shape**

**write** (*fd, header, shape*)

Write to a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**class** `pytoshop.layers.GlobalLayerMaskInfo` (*overlay\_color\_space=b'x00x00x00x00x00x00x00x00x00x00', opacity=100, kind=<LayerMaskKind.use\_value\_stored\_per\_layer: 128>*)

Bases: `object`

Global layer mask info.

**kind**

Layer mask kind. See `enums.LayerMaskKind`

**opacity**

Opacity. 0=transparent, 100=opaque

**overlay\_color\_space**

Undocumented

**classmethod read** (*fd, header*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**write** (*fd, header*)

Write to a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**class** `pytoshop.layers.LayerAndMaskInfo` (*layer\_info=None, global\_layer\_mask\_info=None, additional\_layer\_info=None*)

Bases: `object`

Layer and mask information section.

**additional\_layer\_info**

List of additional layer info. See `TaggedBlock`.

**additional\_layer\_info\_map**

A mapping from tagged block codes to `tagged_block.TaggedBlock` instances.

This is a convenience to more easily get associated tagged blocks.

**global\_layer\_mask\_info**

Global layer mask info. See `GlobalLayerMaskInfo`.

**layer\_info**

Layer info. See `LayerInfo`.

**classmethod read** (*fd*, *header*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**write** (*fd*, *header*)

Write to a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**class** `pytoshop.layers.LayerInfo` (*layer\_records=None*, *use\_alpha\_channel=False*)

Bases: `object`

A set of `LayerRecord` instances.

**layer\_records**

List of `LayerRecord` instances

**classmethod read** (*fd*, *header*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**use\_alpha\_channel**

Indicates that the first channel contains transparency data for the merged result.

**write** (*fd*, *header*)

Write to a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**class** `pytoshop.layers.LayerMask` (*top=0*, *left=0*, *bottom=0*, *right=0*, *default\_color=False*, *position\_relative\_to\_layer=False*, *layer\_mask\_disabled=False*, *invert\_layer\_mask\_when\_blending=False*, *user\_mask\_from\_rendering\_other\_data=False*, *user\_mask\_density=None*, *user\_mask\_feather=None*, *vector\_mask\_density=None*, *vector\_mask\_feather=None*, *real\_flags=0*, *real\_user\_mask\_background=False*, *real\_top=0*, *real\_left=0*, *real\_bottom=0*, *real\_right=0*)

Bases: `object`

Layer mask / adjustment layer data.

**bottom**

Bottom of rectangle enclosing layer mask

**default\_color**

Default color for mask

**height**

Height of the mask layer.

**invert\_layer\_mask\_when\_blending**

Invert layer mask when blending (obsolete)

**layer\_mask\_disabled**

Layer mask disabled

**left**

Left of rectangle enclosing layer mask

**length** (*header*)

The length of the section, in bytes, not including its header.

**position\_relative\_to\_layer**

position relative to layer

**classmethod read** (*fd*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**real\_bottom**

**real\_flags**

**real\_height**

Real height of the mask layer.

**real\_left**

**real\_right**

**real\_shape**

Real shape of the mask layer (*height, width*).

**real\_top**

**real\_user\_mask\_background**

**real\_width**

Real width of the mask layer.

**right**

Right of rectangle enclosing layer mask

**shape**

Shape of the mask layer (*height, width*).

**top**

Top of rectangle enclosing layer mask

**total\_length** (*header*)

The length of the section, in bytes, including its header.

**user\_mask\_density**

User mask density

**user\_mask\_feather**

User mask feather

**user\_mask\_from\_rendering\_other\_data**

Indicates that the user mask actually came from rendering other data.

**vector\_mask\_density**

Vector mask density

**vector\_mask\_feather**

Vector mask feather

**width**

Width of the mask layer.

**write** (*fd, header*)

Write to a file-like object.

#### Parameters

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

```
class pytoshop.layers.LayerRecord(top=0, left=0, bottom=0, right=0,
                                   blend_mode=<BlendMode.normal: b'norm'>, opac-
                                   ity=255, clipping=False, transparency_protected=False,
                                   visible=True, pixel_data_irrelevant=False, name="",
                                   channels=None, blocks=None, color_mode=None)
```

Bases: `object`

Layer record.

There is one of these per logical layer in the file.

**blend\_mode**

Blend mode. See `enums.BlendMode`

**blending\_ranges**

**blocks**

List of `tagged_block.TaggedBlock` items with additional information about this layer.

**blocks\_map**

A mapping from tagged block codes to `tagged_block.TaggedBlock` instances.

This is a convenience to more easily get associated tagged blocks.

**bottom**

Bottom of rectangle enclosing layer

**channels**

Dictionary from `enums.ChannelId` to `ChannelImageData`.

For safety against different color modes, it is better to use `get_channel` and `set_channel`.

**clipping**

Clipping. False=base, True=non-base

**get\_channel** (*color*)

Get a channel for a given color. Raises an error if the color space doesn't have the given color.

**Parameters** **color** (`enums.ColorChannel`) –

**Returns** **channel**

**Return type** *ChannelImageData*

**height**

Height of the layer.

**left**

Left of rectangle enclosing layer

**mask**

**name**

Name of layer

**opacity**

Opacity. 0=transparent, 255=opaque

**pixel\_data\_irrelevant**

Pixel data is irrelevant to appearance of document

**classmethod read** (*fd, header*)

Instantiate from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**read\_channel\_data** (*fd, header*)

Read the *ChannelImageData* for this layer.

**right**

Right of rectangle enclosing layer

**set\_channel** (*color, channel*)

Set a channel for a given color. Raises an error if the color space doesn't have the given color.

**Parameters**

- **color** (`enums.ColorChannel`) –
- **channel** (`ChannelImageData`) –

**shape**

Shape of the layer (*height, width*).

**top**

Top of rectangle enclosing layer

**transparency\_protected**

Transparency protected

**visible**

Visible

**width**

Width of the layer.

**write** (*fd*, *header*)  
Write to a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**write\_channel\_data** (*fd*, *header*)  
Write the *ChannelImageData* for this layer.

## 4.2.7 pytoshop.path

Handle Bézier paths.

**class** `pytoshop.path.ClipboardRecord` (*top=0.0, left=0.0, bottom=0.0, right=0.0, resolution=0*)  
Bases: `pytoshop.path.PathRecord`

**bottom**  
bottom, in pixels

**left**  
left, in pixels

**classmethod** `read_data` (*fd*, *header*)

**resolution**

**right**  
right, in pixels

**top**  
top, in pixels

**write\_data** (*fd*, *header*)

**class** `pytoshop.path.ClosedSubpathBezierKnotLinked` (*y0=0.0, x0=0.0, y1=None, x1=None, y2=None, x2=None*)  
Bases: `pytoshop.path._PointRecord`

**class** `pytoshop.path.ClosedSubpathBezierKnotUnlinked` (*y0=0.0, x0=0.0, y1=None, x1=None, y2=None, x2=None*)  
Bases: `pytoshop.path._PointRecord`

**class** `pytoshop.path.ClosedSubpathLengthRecord` (*num\_knots=0*)  
Bases: `pytoshop.path._LengthRecord`

**class** `pytoshop.path.InitialFillRuleRecord` (*all\_pixels=False*)  
Bases: `pytoshop.path.PathRecord`

**all\_pixels**  
Fill starts with all pixels

**classmethod** `read_data` (*fd*, *header*)

**write\_data** (*fd*, *header*)

**class** `pytoshop.path.OpenSubpathBezierKnotLinked` (*y0=0.0, x0=0.0, y1=None, x1=None, y2=None, x2=None*)  
Bases: `pytoshop.path._PointRecord`

```

class pytoshop.path.OpenSubpathBezierKnotUnlinked (y0=0.0,    x0=0.0,    y1=None,
                                                    x1=None, y2=None, x2=None)
    Bases: pytoshop.path._PointRecord

class pytoshop.path.OpenSubpathLengthRecord (num_knots=0)
    Bases: pytoshop.path._LengthRecord

class pytoshop.path.PathFillRuleRecord
    Bases: pytoshop.path.PathRecord
    classmethod read_data (fd, header)
    write_data (fd, header)

class pytoshop.path.PathRecord
    Bases: object
    length (header)
    classmethod read (fd, header)
    classmethod read_data (fd, header)
    type
    write (fd, header)
    write_data (fd, header)

class pytoshop.path.PathResource (path_records=[])
    Bases: object
    classmethod from_rect (top, left, bottom, right, all_pixels=True)
    length (header)
        The length of the section, in bytes, not including its header.
    path_records
        List of PathRecord instances.
    classmethod read (fd, length, header)
    write (fd, header)

```

## 4.2.8 pytoshop.tagged\_block

*TaggedBlock* objects.

```

class pytoshop.tagged_block.GenericTaggedBlock (code=b", data=b")
    Bases: pytoshop.tagged_block.TaggedBlock
    A generic TaggedBlock subclass for tag codes pytoshop doesn't know about.
    code
    data
    data_length (header)
    classmethod read_data (fd, code, length, header)
    write_data (fd, header)

class pytoshop.tagged_block.LayerColor (color=0)
    Bases: pytoshop.tagged_block.TaggedBlock

```



```

    color
        Color

    data_length (header)

    classmethod read_data (fd, code, length, header)

    write_data (fd, header)

class pytoshop.tagged_block.LayerId (id=0)
    Bases: pytoshop.tagged_block.TaggedBlock

    data_length (header)

    id
        Layer id

    classmethod read_data (fd, code, length, header)

    write_data (fd, header)

class pytoshop.tagged_block.LayerNameSource (id=0)
    Bases: pytoshop.tagged_block.TaggedBlock

    data_length (header)

    id
        The layer id of the source of the name of this layer

    classmethod read_data (fd, code, length, header)

    write_data (fd, header)

class pytoshop.tagged_block.MetadataSetting (datas=None)
    Bases: pytoshop.tagged_block.TaggedBlock

    data_length (header)

    datas

    classmethod read_data (fd, code, length, header)

    write_data (fd, header)

class pytoshop.tagged_block.NestedSectionDividerSetting (type=<SectionDividerSetting.open:
    I>, key=None, sub-
    type=None)

    Bases: pytoshop.tagged_block._SectionDividerSetting

class pytoshop.tagged_block.SectionDividerSetting (type=<SectionDividerSetting.open:
    I>, key=None, subtype=None)

    Bases: pytoshop.tagged_block._SectionDividerSetting

class pytoshop.tagged_block.TaggedBlock
    Bases: object

    code

    data_length (header)

    static is_long_length (code, header)

    length (header)
        The length of the section, in bytes, not including its header.

    classmethod read (fd, header, padding=1)
        Instantiate from a file-like object.

```

#### Parameters

- **fd** (*file-like object*) – Must be readable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**classmethod read\_data** (*fd, code, length, header*)

**total\_length** (*header, padding=1*)

The length of the section, in bytes, including its header.

**write** (*fd, header, padding=1*)

Write to a file-like object.

#### Parameters

- **fd** (*file-like object*) – Must be writable, seekable and open in binary mode.
- **header** (*PsdFile object*) – An object to get global file information from.

**write\_data** (*fd, header*)

**class** `pytoshop.tagged_block.UnicodeLayerName` (*name=""*)

Bases: `pytoshop.tagged_block.TaggedBlock`

**data\_length** (*header*)

**name**

The name of the layer.

**classmethod read\_data** (*fd, code, length, header*)

**write\_data** (*fd, header*)

**class** `pytoshop.tagged_block.VectorMask` (*version=3, invert=False, not\_link=False, disable=False, path\_resource=None*)

Bases: `pytoshop.tagged_block.TaggedBlock`

**data\_length** (*header*)

**disable**

Disable mask

**invert**

Invert mask

**not\_link**

Don't link mask

**path\_resource**

`path.PathResource` instance<sup>4</sup>

**classmethod read\_data** (*fd, code, length, header*)

**version**

Vector mask block version

**write\_data** (*fd, header*)

## 4.3 Enums

---

*enums*

Enumerated values used throughout the library.

---

### 4.3.1 pytoshop.enums

Enumerated values used throughout the library.

```
class pytoshop.enums.BlendMode
```

```
    Bases: bytes, enum.Enum
```

```
    Layer blend mode.
```

```
    color = b'colr'
```

```
    color_burn = b'idiv'
```

```
    color_dodge = b'div '
```

```
    darken = b'dark'
```

```
    darker_color = b'dkCl'
```

```
    difference = b'diff'
```

```
    dissolve = b'diss'
```

```
    divide = b'fdiv'
```

```
    exclusion = b'smud'
```

```
    hard_light = b'hLit'
```

```
    hard_mix = b'hMix'
```

```
    hue = b'hue '
```

```
    lighten = b'lite'
```

```
    lighter_color = b'lgCl'
```

```
    linear_burn = b'lbrn'
```

```
    linear_dodge = b'lddg'
```

```
    linear_light = b'lLit'
```

```
    luminosity = b'lum '
```

```
    multiply = b'mul '
```

```
    normal = b'norm'
```

```
    overlay = b'over'
```

```
    pass_through = b'pass'
```

```
    pin_light = b'pLit'
```

```
    saturation = b'sat '
```

```
    screen = b'scrn'
```

```
    soft_light = b'sLit'
```

```
    subtract = b'fsub'
```

```
    vivid_light = b'vLit'
```

```
class pytoshop.enums.ChannelId
```

```
    Bases: enum.IntEnum
```

```
    Channel id.
```

Used to map channel data to image planes in layers.

The meaning of the positive numbers depends on the *ColorMode* in effect.

```
L = 0
a = 1
b = 2
bitmap = 0
black = 3
blue = 2
cyan = 0
default = 0
gray = 0
green = 1
magenta = 1
real_user_layer_mask = -3
red = 0
transparency = -1
user_layer_mask = -2
yellow = 2
```

**class** `pytoshop.enums.ColorChannel`

Bases: `enum.IntEnum`

Color channel names, with unique values so we can check that they apply to the expected color mode.

```
L = 10
a = 11
b = 12
bitmap = 0
black = 9
blue = 5
cyan = 6
default = 2
gray = 1
green = 4
magenta = 7
real_user_layer_mask = -3
red = 3
transparency = -1
user_layer_mask = -2
```

```
yellow = 8
```

```
class pytoshop.enums.ColorDepth
```

```
Bases: enum.IntEnum
```

Color depth (bits-per-pixel-per-channel).

Supported values are 1, 8, 16, and 32.

```
depth1 = 1
```

```
depth16 = 16
```

```
depth32 = 32
```

```
depth8 = 8
```

```
class pytoshop.enums.ColorMode
```

```
Bases: enum.IntEnum
```

Color mode.

```
bitmap = 0
```

```
cmyk = 4
```

```
duotone = 8
```

```
grayscale = 1
```

```
indexed = 2
```

```
lab = 9
```

```
multichannel = 7
```

```
rgb = 3
```

```
class pytoshop.enums.ColorSpace
```

```
Bases: enum.IntEnum
```

Color space ids.

Defines the meaning of the 4 color values in a color structure.

- `rgb`: The first three values are *red*, *green*, and *blue*. The are full 16-bit unsigned values as in Apple's `RGBColor` data structure.
- `hsb`: The first three values are *hue*, *saturation*, and *brightness*. The are full unsigned 16-bit values as in Apple's `HSVColor` data structure.
- `cmyk`: The four values are *cyan*, *magenta*, *yellow* and *black*. The are full unsigned 16-bit values. 0 = 100% ink.
- `lab`: The first three values are *lightness*, *a chrominance*, and *b chrominance*. Lightness is a 16-bit value from 0 to 10000. Chrominance components are each 16-bit values from -12800 to 12700. Gray values are represented by chrominance components of 0.
- `grayscale`: The first value in the color data is the gray value from 0 to 10000.

Additional values are for “custom color spaces” which are not as well documented:

- `pantone`: Pantone® matching system.
- `focoltone`: Focoltone matching system.
- `trumatch`: Trumatch color.
- `toyo_88_colorfinder_1050`: Toyo 88 colorfinder 1050.

- `hks`: HKS colors.

```
cmypk = 2
focoltone = 4
grayscale = 8
hks = 10
hsb = 1
lab = 7
pantone = 3
rgb = 0
toyo_88_colorfinder_1050 = 6
trumatch = 5
```

**class** `pytoshop.enums.Compression`

Bases: `enum.IntEnum`

Compression mode.

- `raw`: raw image data.
- `rle`: Run length encoded (RLE) compressed. The RLE compression is the same compression algorithm used by the Macintosh ROM routine PackBits, and the TIFF standard.
- `zip`: Zip (zlib) without prediction.
- `zip_prediction`: Zip (zlib) with prediction.

```
raw = 0
rle = 1
zip = 2
zip_prediction = 3
```

**class** `pytoshop.enums.GuideDirection`

Bases: `enum.IntEnum`

An enumeration.

```
horizontal = 1
vertical = 0
```

**class** `pytoshop.enums.ImageResourceID`

Bases: `enum.IntEnum`

Ids for Image Resource blocks.

```
alpha_channel_names = 1006
alpha_identifiers = 1053
alternate_duotone_colors = 1066
alternate_spot_colors = 1067
auto_save_file_path = 1086
auto_save_format = 1087
```

```
background_color = 1010
border_info = 1009
caption = 1008
caption_digest = 1061
color_halftoning_info = 1013
color_samplers_resource = 1073
color_transfer_funcs = 1016
copyright_flag = 1034
count_info = 1080
display_info = 1077
document_specific_ids_seed_number = 1044
duotone_halftoning_info = 1014
duotone_image_info = 1018
duotone_transfer_funcs = 1017
effective_black_and_white = 1019
effects_visible = 1042
eps_options = 1021
exif_data_1 = 1058
exif_data_2 = 1059
global_altitude = 1049
global_angle = 1037
grayscale_and_multichannel_halftoning_info = 1012
grayscale_and_multichannel_transfer_func = 1015
grid_and_guides_info = 1032
hdr_toning_info = 1070
icc_profile = 1039
icc_untagged_profile = 1041
image_mode_for_raw = 1029
image_ready_7_rollover_expanded_state = 7003
image_ready_data_sets = 7001
image_ready_default_selected_state = 7002
image_ready_rollover_expanded_state = 7004
image_ready_save_layer_settings = 7005
image_ready_variables = 7000
image_ready_version = 7006
indexed_color_table_count = 1046
```

```
iptc_naa = 1028
jpeg_quality = 1030
jump_to_xpex = 1052
layer_comps = 1065
layer_groups_enabled = 1072
layer_selection_ids = 1069
layer_state_info = 1024
layers_group_info = 1026
lightroom_workflow = 8000
mac_nsprintinfo = 1084
mac_page_format_info = 1002
mac_print_info = 1001
measurement_scale = 1074
name_of_clipping_path = 2999
onion_skins = 1078
origin_path_info = 3000
path_selection_state = 1088
pixel_aspect_ratio = 1064
print_flags = 1011
print_flags_info = 10000
print_info = 1071
print_info_cs5 = 1082
print_scale = 1062
print_style = 1083
quick_mask_info = 1022
resolution_info = 1005
sheet_disclosure = 1076
slices = 1050
spot_halftone = 1043
thumbnail_resource = 1036
timeline_info = 1075
transparency_index = 1047
unicode_alpha_names = 1045
url = 1035
url_list = 1054
version_info = 1057
```



```

watermark = 1040
win_devmode = 1085
workflow_url = 1051
xmp_metadata = 1060

```

```
class pytoshop.enums.LayerMaskKind
```

```
Bases: enum.IntEnum
```

Layer mask kind.

According to the spec, only `use_value_stored_per_layer` is preferred. The other are retained for backward compatibility only.

```

color_protected = 1
color_selected = 0
use_value_stored_per_layer = 128

```

```
class pytoshop.enums.PathRecordType
```

```
Bases: enum.IntEnum
```

An enumeration.

```

clipboard_record = 7
closed_subpath_bezier_knot_linked = 1
closed_subpath_bezier_knot_unlinked = 2
closed_subpath_length = 0
initial_fill_rule_record = 8
open_subpath_bezier_knot_linked = 4
open_subpath_bezier_knot_unlinked = 5
open_subpath_length = 3
path_fill_rule_record = 6

```

```
class pytoshop.enums.PrintScaleStyle
```

```
Bases: enum.IntEnum
```

An enumeration.

```

centered = 0
size_to_fit = 1
user_defined = 2

```

```
class pytoshop.enums.SectionDividerSetting
```

```
Bases: enum.IntEnum
```

Section divider setting.

Used for the mode of grouped layers.

- `any_other`: any other type of layer
- `open`: Display an open folder
- `closed`: Display a closed folder
- `bounding`: Bounding section divider, hidden in GUI.

```

any_other = 0
bounding = 3
closed = 2
open = 1

```

**class** pytoshop.enums.Units

Bases: `enum.IntEnum`

An enumeration.

```

cm = 2
columns = 5
inches = 1
picas = 4
points = 3

```

**class** pytoshop.enums.Version

Bases: `enum.IntEnum`

The PSD file version.

Version 1 is the classic “PSD” file.

Version 2 is the large document format “PSB” file which supports documents up to 300,000 pixels in any dimension.

```

psb = 2
psd = 1
version_1 = 1
version_2 = 2

```

## 4.4 Codecs

---

*codecs*

Coders and decoders for the various compression types in PSD.

---

### 4.4.1 pytoshop.codecs

Coders and decoders for the various compression types in PSD.

`pytoshop.codecs.compress_constant_raw` (*fd, value, width, rows, depth, version*)

Write a virtual image containing a constant to a raw stream.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **value** (*int*) – The constant value in the generated virtual image.
- **width** (*int*) – The width of the image, in pixels.
- **rows** (*int*) – The number of rows in the image, in pixels. This is `height * num_channels`.

- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.compress_constant_rle` (*fd, value, width, rows, depth, version*)

Write a virtual image containing a constant to a runlength-encoded stream.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **value** (*int*) – The constant value in the generated virtual image.
- **width** (*int*) – The width of the image, in pixels.
- **rows** (*int*) – The number of rows in the image, in pixels. This is `height * num_channels`.
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.compress_constant_zip` (*fd, value, width, rows, depth, version*)

Write a virtual image containing a constant to a zip compressed stream.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **value** (*int*) – The constant value in the generated virtual image.
- **width** (*int*) – The width of the image, in pixels.
- **rows** (*int*) – The number of rows in the image, in pixels. This is `height * num_channels`.
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.compress_constant_zip_prediction` (*fd, value, width, rows, depth, version*)

Write a virtual image containing a constant to a zip with prediction compressed stream.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **value** (*int*) – The constant value in the generated virtual image.
- **width** (*int*) – The width of the image, in pixels.
- **rows** (*int*) – The number of rows in the image, in pixels. This is `height * num_channels`.
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.compress_image` (*fd, image, compression, shape, num\_channels, depth, version*)

Write an image with the given compression type.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **image** (*2-D numpy array or scalar*) – The image to compress. Must be unsigned integer with 8, 16 or 32 bits. If *depth* is 1, the array should have dtype `uint8` with a byte per pixel. If a scalar, a “virtual” image will be used as if the image contained only that constant value.
- **compression** (`enums.Compression`) – The compression format to use. See `enums.Compression`.
- **shape** (*2-tuple of int*) – The shape of the image (*height*, *width*). If *image* is an array, the *shape* is used to confirm it has the correct shape. If *image* is a constant, *shape* is used to generate the virtual constant image.
- **num\_channels** (*int*) – The number of color channels in the image. If *image* is an array, the *num\_channels* is used to confirm it has the correct number of channels. If *image* is a constant, *num\_channels* is used to generate the virtual constant image.
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`. If *image* is an array, the *depth* is used to confirm it has the correct number of channels. If *image* is a constant, *depth* is used to generate the virtual constant image.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.compress_raw` (*fd, image, depth, version*)

Write a Numpy array to raw bytes in a file.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **image** (*2-D numpy array*) – The image to compress. Must be unsigned integer with 8, 16 or 32 bits. If *depth* is 1, the array should have dtype `uint8` with a byte per pixel.
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.compress_rle` (*fd, image, depth, version*)

Write a Numpy array to a run length encoded stream.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **image** (*2-D numpy array*) – The image to compress. Must be unsigned integer with 8, 16 or 32 bits. If *depth* is 1, the array should have dtype `uint8` with a byte per pixel.
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.compress_zip` (*fd, image, depth, version*)

Write a Numpy array to a zip (zlib) compressed stream.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **image** (*2-D numpy array*) – The image to compress. Must be unsigned integer with 8, 16 or 32 bits. If *depth* is 1, the array should have dtype `uint8` with a byte per pixel.

- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.compress_zip_prediction` (*fd, image, depth, version*)

Write a Numpy array to a zip (zlib) with prediction compressed stream.

Not supported for 1- or 32-bit images.

#### Parameters

- **fd** (*file-like object*) – Writable file-like object, open in binary mode.
- **image** (*2-D numpy array*) – The image to compress. Must be unsigned integer with 8, 16 or 32 bits. If *depth* is 1, the array should have dtype `uint8` with a byte per pixel.
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

`pytoshop.codecs.decompress_image` (*data, compression, shape, depth, version*)

Decompress data with the given compression.

#### Parameters

- **data** (*bytes*) – The raw bytes from the file.
- **compression** (`enums.Compression`) – The compression format to use. See `enums.Compression`.
- **shape** (*2-tuple of int*) – The shape of the resulting array, (height, width).
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

**Returns image** – The image data as a Numpy array.

**Return type** numpy array

`pytoshop.codecs.decompress_raw` (*data, shape, depth, version*)

Converts raw data to a Numpy array.

#### Parameters

- **data** (*bytes*) – The raw bytes from the file.
- **shape** (*2-tuple of int*) – The shape of the resulting array, (height, width).
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

**Returns image** – The image data as a Numpy array. If *depth* is 1, the array is expanded to `uint8` with a byte per pixel.

**Return type** numpy array

`pytoshop.codecs.decompress_rle` (*data, shape, depth, version*)

Decompress run length encoded data.

#### Parameters

- **data** (*bytes*) – The raw bytes from the file.

- **shape** (*2-tuple of int*) – The shape of the resulting array, (*height*, *width*).
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

**Returns image** – The image data as a Numpy array. If *depth* is 1, the array is expanded to `uint8` with a byte per pixel.

**Return type** numpy array

`pytoshop.codecs.decompress_zip` (*data*, *shape*, *depth*, *version*)  
Decompress zip (zlib) encoded data.

**Parameters**

- **data** (*bytes*) – The raw bytes from the file.
- **shape** (*2-tuple of int*) – The shape of the resulting array, (*height*, *width*).
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

**Returns image** – The image data as a Numpy array. If *depth* is 1, the array is expanded to `uint8` with a byte per pixel.

**Return type** numpy array

`pytoshop.codecs.decompress_zip_prediction` (*data*, *shape*, *depth*, *version*)  
Decompress zip (zlib) with prediction encoded data.

Not supported for 1- or 32-bit images.

**Parameters**

- **data** (*bytes*) – The raw bytes from the file.
- **shape** (*2-tuple of int*) – The shape of the resulting array, (*height*, *width*).
- **depth** (`enums.ColorDepth`) – The bit depth of the image. See `enums.ColorDepth`.
- **version** (`enums.Version`) – The version of the PSD file. See `enums.Version`.

**Returns image** – The image data as a Numpy array. If *depth* is 1, the array is expanded to `uint8` with a byte per pixel.

**Return type** numpy array

`pytoshop.codecs.normalize_image` (*image*, *depth*)

## 4.5 Utilities

---

*util*

Miscellaneous utilities.

---

### 4.5.1 pytoshop.util

Miscellaneous utilities.

`pytoshop.util.assert_is_list_of` (*value*, *cls*, *min=None*, *max=None*)

If value is not a list of cls instances, raises TypeError.

`pytoshop.util.decode_unicode_string(data)`  
Decode Photoshop's definition of a [Unicode String](#).

`pytoshop.util.do_byteswap(arr)`  
Return a copy of an array, byteswapped.

`pytoshop.util.encode_unicode_string(s)`  
Encode Photoshop's definition of a [Unicode String](#).

`pytoshop.util.ensure_bigendian(arr)`  
Ensure that a Numpy array is in big-endian order.  
  
Returns a copy if the endianness needed to be changed.

`pytoshop.util.ensure_native_endian(arr)`  
Ensure that a Numpy array is in native-endian order.  
  
Returns a copy if the endianness needed to be changed.

`pytoshop.util.get_channel(color, color_mode, channels)`

`pytoshop.util.log(msg, *args)`  
Print a logging message if debugging is turned on.

`pytoshop.util.needs_byteswap(arr)`  
Returns True if the array needs to be byteswapped.

`pytoshop.util.pack_bitflags(*values)`  
Pack separate booleans back into a bit field.

`pytoshop.util.pad(number, divisor)`  
Pads an integer up to the given divisor.

`pytoshop.util.pascal_string_length(value, padding=1)`  
Calculates the total length of writing a UTF-8-encoded Pascal string to disk.

**Parameters** `value` (*str*) – A unicode string value.

**Returns** `length` – The length, in bytes.

**Return type** `int`

`pytoshop.util.read_pascal_string(fd, padding=1)`  
Read a UTF-8-encoded Pascal string from a file.

**Parameters**

- `fd` (*file-like object*) – Must be opened for reading, seekable and in binary mode.
- `padding` (*int, optional*) – If provided, additional pad bytes will be read until the total amount read is a multiple of padding.

**Returns** `value` – The unicode value of the string.

**Return type** `str`

`pytoshop.util.read_unicode_string(fd)`  
Read a UTF-16-BE-encoded Unicode string (with length) from a file.

**Parameters** `fd` (*file-like object*) – Must be opened for reading, seekable and in binary mode.

**Returns** `value` – The unicode value of the string.

**Return type** `str`

`pytoshop.util.read_value` (*fd*, *fmt*, *endian*='>')

Read a values from a file-like object.

**Parameters**

- **fd** (*file-like object*) – Must be opened for reading, in binary mode.
- **fmt** (*str*) – A `struct` module `format character` string.
- **endian** (*str*) – The endianness. Must be > or <. Default: >.

**Returns**

**value** – The value(s) read from the file.

If a single value, it is returned alone. If multiple values, a tuple is returned.

**Return type** any

`pytoshop.util.set_channel` (*color*, *channel*, *color\_mode*, *channels*)

`pytoshop.util.trace_read` (*func*)

Prints debugging information from a read or write method.

For internal use only.

`pytoshop.util.trace_write` (*func*)

Prints debugging information from a read or write method.

For internal use only.

`pytoshop.util.unicode_string_length` (*value*)

Calculates the total length of writing a UTF-16-BE-encoded Unicode string (with length) to a file.

**Parameters** **value** (*str*) – A unicode string value.

**Returns** **length** – The length, in bytes.

**Return type** int

`pytoshop.util.unpack_bitflags` (*value*, *nbits*)

Unpack a bitfield into its constituent parts.

`pytoshop.util.write_pascal_string` (*fd*, *value*, *padding*=1)

Write a UTF-8-encoded Pascal string to a file.

**Parameters**

- **fd** (*file-like object*) – Must be opened for writing and in binary mode.
- **value** (*str*) – A unicode string value.
- **padding** (*int*, *optional*) – If provided, additional pad bytes will be written until the total amount written is a multiple of padding.

`pytoshop.util.write_unicode_string` (*fd*, *value*)

Write a UTF-16-BE-encoded Unicode string (with length) to a file.

**Parameters**

- **fd** (*file-like object*) – Must be opened for writing and in binary mode.
- **value** (*str*) – A unicode string value.

`pytoshop.util.write_value` (*fd*, *fmt*, *\*value*, *\*\*kwargs*)

Write a single binary value to a file-like object.

**Parameters**



- **fd** (*file-like object*) – Must be opened for writing, in binary mode.
- **fmt** (*str*) – A `struct` module `format character` string.
- **value** (*any*) – The value to encode and write to the file.
- **endian** (*str*) – The endianness. Must be `>` or `<`. Default: `>`.



### 5.1 Development Lead

- Michael Droettboom <[mdboom@gmail.com](mailto:mdboom@gmail.com)>

Parts of this code were inspired by `psd-tools`, so a huge thanks to Mikhail Korobov and the rest of the `psd-tools` team.

### 5.2 Contributors



### 6.1 1.2.1 (2018-11-30) (2017-12-13) (2017-09-26)

Bugfixes:

- Fix #4: Change masked layer to same size as image layer when using the *user.nested\_layers* interface.

### 6.2 1.0.1 (2017-08-01) (2017-07-29)

- Declared API as stable.

### 6.3 0.6.0 (2017-07-24)

Minor improvements:

- Make it easier to choose the right channel, given the current color mode.

### 6.4 0.5.0 (2017-07-03)

Minor improvements:

- The color assigned to a layer (displayed in the layer list) is now available in the *nested\_layers* API.

Bugfixes:

- Fix crash in `GuideResourceBlock`.

## 6.5 0.4.1 (2017-06-05)

Minor improvements:

- Parse multiple values in a struct together when possible. This has a modest speed improvement.

Bugfixes:

- Fix pip install [#1]
- The main canvas image will use the specified compression algorithm when using `nested_layers_to_psd`.
- Non-image layers correctly set “`pixel_data_is_irrelevant`” flag.

## 6.6 0.4.0 (2017-05-12)

Improvements:

- For speed purposes, `pytoshop` no longer uses traitlets.
- Performance improvements to the compression/decompression code.
- Added support for the `shmd` metadata tagged block, and the ability to access it from the `user.nested_layers` API.

Bugfixes:

- Updated the list of tagged blocks that use 8-bit lengths.
- Fixed a bug where the image data would be corrupted when writing images from an input file to an output file with a different file format version.
- Fixed a crash when the input file contains no layer group ids.
- Allow Numpy arrays of shape `()` in place of scalars for constant images.

## 6.7 0.3.0 (2017-01-09)

Improvements:

- `pytoshop` now runs on Python 2.7, in addition to 3.4 and 3.5.
- Many of the image resources types are now handled directly, rather than through a generic bytes-only interface.
- Major speedups in compression codecs.

Bugfixes:

- Saving a layer with a constant color (in `nested_layers_to_psd`) now works correctly.
- Unicode string decoding now properly handles trailing zeroes.
- The “name source” on layers (when created from `nested_layers_to_psd`) would point to the wrong source, but is now fixed.
- Fix a bug when writing a layer of width 1.

### p

- `pytoshop.blending_range`, 12
- `pytoshop.codecs`, 38
- `pytoshop.color_mode`, 15
- `pytoshop.core`, 13
- `pytoshop.enums`, 31
- `pytoshop.image_data`, 15
- `pytoshop.image_resources`, 16
- `pytoshop.layers`, 21
- `pytoshop.path`, 27
- `pytoshop.tagged_block`, 28
- `pytoshop.user.nested_layers`, 9
- `pytoshop.util`, 42





## A

- a (pytoshop.enums.ChannelId attribute), 32
- a (pytoshop.enums.ColorChannel attribute), 32
- additional\_layer\_info (pytoshop.layers.LayerAndMaskInfo attribute), 22
- additional\_layer\_info\_map (pytoshop.layers.LayerAndMaskInfo attribute), 22
- all\_pixels (pytoshop.path.InitialFillRuleRecord attribute), 27
- alpha\_channel\_names (pytoshop.enums.ImageResourceID attribute), 34
- alpha\_identifiers (pytoshop.enums.ImageResourceID attribute), 34
- AlphaIdentifiers (class in pytoshop.image\_resources), 16
- alternate\_duotone\_colors (pytoshop.enums.ImageResourceID attribute), 34
- alternate\_spot\_colors (pytoshop.enums.ImageResourceID attribute), 34
- altitude (pytoshop.image\_resources.GlobalAltitude attribute), 17
- angle (pytoshop.image\_resources.GlobalAngle attribute), 17
- any\_other (pytoshop.enums.SectionDividerSetting attribute), 37
- assert\_is\_list\_of() (in module pytoshop.util), 42
- auto\_save\_file\_path (pytoshop.enums.ImageResourceID attribute), 34
- auto\_save\_format (pytoshop.enums.ImageResourceID attribute), 34
- attribute), 34
- BackgroundColor (class in pytoshop.image\_resources), 16
- base\_value (pytoshop.image\_resources.DocumentSpecificIdsSeedNumber attribute), 17
- bitmap (pytoshop.enums.ChannelId attribute), 32
- bitmap (pytoshop.enums.ColorChannel attribute), 32
- bitmap (pytoshop.enums.ColorMode attribute), 33
- black (pytoshop.enums.ChannelId attribute), 32
- black (pytoshop.enums.ColorChannel attribute), 32
- black0 (pytoshop.blending\_range.BlendingRange attribute), 12
- black1 (pytoshop.blending\_range.BlendingRange attribute), 12
- blend\_mode (pytoshop.layers.LayerRecord attribute), 25
- blend\_mode (pytoshop.user.nested\_layers.Layer attribute), 10
- blending\_ranges (pytoshop.layers.LayerRecord attribute), 25
- BlendingRange (class in pytoshop.blending\_range), 12
- BlendingRangePair (class in pytoshop.blending\_range), 12
- BlendingRanges (class in pytoshop.blending\_range), 13
- BlendMode (class in pytoshop.enums), 31
- blocks (pytoshop.image\_resources.ImageResources attribute), 19
- blocks (pytoshop.layers.LayerRecord attribute), 25
- blocks\_map (pytoshop.layers.LayerRecord attribute), 25
- blue (pytoshop.enums.ChannelId attribute), 32
- blue (pytoshop.enums.ColorChannel attribute), 32
- border\_info (pytoshop.enums.ImageResourceID attribute), 35
- border\_width\_den (pytoshop.image\_resources.BorderInfo attribute), 16
- border\_width\_num (pytoshop.image\_resources.BorderInfo attribute), 16
- BorderInfo (class in pytoshop.image\_resources), 16
- bottom (pytoshop.layers.LayerMask attribute), 24
- bottom (pytoshop.layers.LayerRecord attribute), 25

## B

- b (pytoshop.enums.ChannelId attribute), 32
- b (pytoshop.enums.ColorChannel attribute), 32
- background\_color (pytoshop.enums.ImageResourceID

- bottom (pytoshop.path.ClipboardRecord attribute), 27
  - bottom (pytoshop.user.nested\_layers.Image attribute), 10
  - bounding (pytoshop.enums.SectionDividerSetting attribute), 38
- C**
- caption (pytoshop.enums.ImageResourceID attribute), 35
  - caption (pytoshop.image\_resources.PrintFlags attribute), 20
  - caption\_digest (pytoshop.enums.ImageResourceID attribute), 35
  - centered (pytoshop.enums.PrintScaleStyle attribute), 37
  - ChannelId (class in pytoshop.enums), 31
  - ChannelImageData (class in pytoshop.layers), 21
  - channels (pytoshop.blending\_range.BlendingRanges attribute), 13
  - channels (pytoshop.image\_data.ImageData attribute), 15
  - channels (pytoshop.layers.LayerRecord attribute), 25
  - channels (pytoshop.user.nested\_layers.Image attribute), 10
  - clipboard\_record (pytoshop.enums.PathRecordType attribute), 37
  - ClipboardRecord (class in pytoshop.path), 27
  - clipping (pytoshop.layers.LayerRecord attribute), 25
  - closed (pytoshop.enums.SectionDividerSetting attribute), 38
  - closed (pytoshop.user.nested\_layers.Group attribute), 9
  - closed\_subpath\_bezier\_knot\_linked (pytoshop.enums.PathRecordType attribute), 37
  - closed\_subpath\_bezier\_knot\_unlinked (pytoshop.enums.PathRecordType attribute), 37
  - closed\_subpath\_length (pytoshop.enums.PathRecordType attribute), 37
  - ClosedSubpathBezierKnotLinked (class in pytoshop.path), 27
  - ClosedSubpathBezierKnotUnlinked (class in pytoshop.path), 27
  - ClosedSubpathLengthRecord (class in pytoshop.path), 27
  - cm (pytoshop.enums.Units attribute), 38
  - cmyk (pytoshop.enums.ColorMode attribute), 33
  - cmyk (pytoshop.enums.ColorSpace attribute), 34
  - code (pytoshop.tagged\_block.GenericTaggedBlock attribute), 28
  - code (pytoshop.tagged\_block.TaggedBlock attribute), 29
  - color (pytoshop.enums.BlendMode attribute), 31
  - color (pytoshop.image\_resources.BackgroundColor attribute), 16
  - color (pytoshop.tagged\_block.LayerColor attribute), 28
  - color\_bars (pytoshop.image\_resources.PrintFlags attribute), 20
  - color\_burn (pytoshop.enums.BlendMode attribute), 31
  - color\_dodge (pytoshop.enums.BlendMode attribute), 31
  - color\_half\_toning\_info (pytoshop.enums.ImageResourceID attribute), 35
  - color\_mode (pytoshop.core.Header attribute), 13
  - color\_mode (pytoshop.user.nested\_layers.Image attribute), 10
  - color\_mode\_data (pytoshop.core.PsdFile attribute), 14
  - color\_protected (pytoshop.enums.LayerMaskKind attribute), 37
  - color\_samplers\_resource (pytoshop.enums.ImageResourceID attribute), 35
  - color\_selected (pytoshop.enums.LayerMaskKind attribute), 37
  - color\_space (pytoshop.image\_resources.BackgroundColor attribute), 16
  - color\_transfer\_funcs (pytoshop.enums.ImageResourceID attribute), 35
  - ColorChannel (class in pytoshop.enums), 32
  - ColorDepth (class in pytoshop.enums), 33
  - ColorMode (class in pytoshop.enums), 33
  - ColorModeData (class in pytoshop.color\_mode), 15
  - ColorSpace (class in pytoshop.enums), 33
  - columns (pytoshop.enums.Units attribute), 38
  - composite\_gray\_blend (pytoshop.blending\_range.BlendingRanges attribute), 13
  - compress\_constant\_raw() (in module pytoshop.codecs), 38
  - compress\_constant\_rle() (in module pytoshop.codecs), 39
  - compress\_constant\_zip() (in module pytoshop.codecs), 39
  - compress\_constant\_zip\_prediction() (in module pytoshop.codecs), 39
  - compress\_image() (in module pytoshop.codecs), 39
  - compress\_raw() (in module pytoshop.codecs), 40
  - compress\_rle() (in module pytoshop.codecs), 40
  - compress\_zip() (in module pytoshop.codecs), 40
  - compress\_zip\_prediction() (in module pytoshop.codecs), 41
  - Compression (class in pytoshop.enums), 34
  - compression (pytoshop.image\_data.ImageData attribute), 15
  - compression (pytoshop.layers.ChannelImageData attribute), 21
  - copyright (pytoshop.image\_resources.CopyrightFlag attribute), 17
  - copyright\_flag (pytoshop.enums.ImageResourceID attribute), 35
  - CopyrightFlag (class in pytoshop.image\_resources), 16
  - count\_info (pytoshop.enums.ImageResourceID attribute), 35
  - crop\_marks (pytoshop.image\_resources.PrintFlags

- attribute), 20
  - cyan (pytoshop.enums.ChannelId attribute), 32
  - cyan (pytoshop.enums.ColorChannel attribute), 32
- D**
- darken (pytoshop.enums.BlendMode attribute), 31
  - darker\_color (pytoshop.enums.BlendMode attribute), 31
  - data (pytoshop.color\_mode.ColorModeData attribute), 15
  - data (pytoshop.image\_resources.GenericImageResourceBlock attribute), 17
  - data (pytoshop.tagged\_block.GenericTaggedBlock attribute), 28
  - data\_length() (pytoshop.image\_resources.AlphaIdentifiers method), 16
  - data\_length() (pytoshop.image\_resources.BackgroundColor method), 16
  - data\_length() (pytoshop.image\_resources.BorderInfo method), 16
  - data\_length() (pytoshop.image\_resources.CopyrightFlag method), 17
  - data\_length() (pytoshop.image\_resources.DocumentSpecificIdsSeedNumber method), 17
  - data\_length() (pytoshop.image\_resources.EffectsVisible method), 17
  - data\_length() (pytoshop.image\_resources.GenericImageResourceBlock method), 17
  - data\_length() (pytoshop.image\_resources.GlobalAltitude method), 17
  - data\_length() (pytoshop.image\_resources.GlobalAngle method), 18
  - data\_length() (pytoshop.image\_resources.GridAndGuidesInfo method), 18
  - data\_length() (pytoshop.image\_resources.GuideResourceBlock method), 18
  - data\_length() (pytoshop.image\_resources.ImageResourceBlock method), 18
  - data\_length() (pytoshop.image\_resources.ImageResourceUnicodeString method), 19
  - data\_length() (pytoshop.image\_resources.LayersGroupInfo method), 20
  - data\_length() (pytoshop.image\_resources.PrintFlags method), 20
  - data\_length() (pytoshop.image\_resources.PrintScale method), 20
  - data\_length() (pytoshop.image\_resources.Url method), 21
  - data\_length() (pytoshop.image\_resources.VersionInfo method), 21
  - data\_length() (pytoshop.tagged\_block.GenericTaggedBlock method), 28
  - data\_length() (pytoshop.tagged\_block.LayerColor method), 29
  - data\_length() (pytoshop.tagged\_block.LayerId method), 29
  - data\_length() (pytoshop.tagged\_block.LayerNameSource method), 29
  - data\_length() (pytoshop.tagged\_block.MetadataSetting method), 29
  - data\_length() (pytoshop.tagged\_block.TaggedBlock method), 29
  - data\_length() (pytoshop.tagged\_block.UnicodeLayerName method), 30
  - data\_length() (pytoshop.tagged\_block.VectorMask method), 30
  - datas (pytoshop.tagged\_block.MetadataSetting attribute), 29
  - decode\_unicode\_string() (in module pytoshop.util), 43
  - decompress\_image() (in module pytoshop.codecs), 41
  - decompress\_raw() (in module pytoshop.codecs), 41
  - decompress\_rle() (in module pytoshop.codecs), 41
  - decompress\_zip() (in module pytoshop.codecs), 42
  - decompress\_zip\_prediction() (in module pytoshop.codecs), 42
  - default (pytoshop.enums.ChannelId attribute), 32
  - default (pytoshop.enums.ColorChannel attribute), 32
  - default\_color (pytoshop.layers.LayerMask attribute), 24
  - depth (pytoshop.core.Header attribute), 14
  - depth1 (pytoshop.enums.ColorDepth attribute), 33
  - depth6 (pytoshop.enums.ColorDepth attribute), 33
  - depth32 (pytoshop.enums.ColorDepth attribute), 33
  - depth8 (pytoshop.enums.ColorDepth attribute), 33
  - difference (pytoshop.enums.BlendMode attribute), 31
  - direction (pytoshop.image\_resources.GuideResourceBlock attribute), 18
  - disable (pytoshop.tagged\_block.VectorMask attribute), 30
  - display\_info (pytoshop.enums.ImageResourceID attribute), 35
  - dissolve (pytoshop.enums.BlendMode attribute), 31
  - divide (pytoshop.enums.BlendMode attribute), 31
  - do\_byteswap() (in module pytoshop.util), 43
  - document\_specific\_ids\_seed\_number (pytoshop.enums.ImageResourceID attribute), 35
  - DocumentSpecificIdsSeedNumber (class in pytoshop.image\_resources), 17
  - dst (pytoshop.blending\_range.BlendingRangePair attribute), 12
  - dtype (pytoshop.layers.ChannelImageData attribute), 21
  - duotone (pytoshop.enums.ColorMode attribute), 33
  - duotone\_half\_toning\_info (pytoshop.enums.ImageResourceID attribute), 35
  - duotone\_image\_info (pytoshop.enums.ImageResourceID attribute), 35
  - duotone\_transfer\_funcs (pytoshop.enums.ImageResourceID attribute), 35

## E

effective\_black\_and\_white (pytoshop.enums.ImageResourceID attribute), 35

effects\_visible (pytoshop.enums.ImageResourceID attribute), 35

EffectsVisible (class in pytoshop.image\_resources), 17

encode\_unicode\_string() (in module pytoshop.util), 43

ensure\_bigendian() (in module pytoshop.util), 43

ensure\_native\_endian() (in module pytoshop.util), 43

eps\_options (pytoshop.enums.ImageResourceID attribute), 35

exclusion (pytoshop.enums.BlendMode attribute), 31

exif\_data\_1 (pytoshop.enums.ImageResourceID attribute), 35

exif\_data\_2 (pytoshop.enums.ImageResourceID attribute), 35

## F

file\_version (pytoshop.image\_resources.VersionInfo attribute), 21

flip (pytoshop.image\_resources.PrintFlags attribute), 20

focoltone (pytoshop.enums.ColorSpace attribute), 34

from\_rect() (pytoshop.path.PathResource class method), 28

## G

GenericImageResourceBlock (class in pytoshop.image\_resources), 17

GenericTaggedBlock (class in pytoshop.tagged\_block), 28

get\_block() (pytoshop.image\_resources.ImageResources method), 19

get\_channel() (in module pytoshop.util), 43

get\_channel() (pytoshop.layers.LayerRecord method), 25

get\_channel() (pytoshop.user.nested\_layers.Image method), 10

global\_altitude (pytoshop.enums.ImageResourceID attribute), 35

global\_angle (pytoshop.enums.ImageResourceID attribute), 35

global\_layer\_mask\_info (pytoshop.layers.LayerAndMaskInfo attribute), 23

GlobalAltitude (class in pytoshop.image\_resources), 17

GlobalAngle (class in pytoshop.image\_resources), 17

GlobalLayerMaskInfo (class in pytoshop.layers), 22

gray (pytoshop.enums.ChannelId attribute), 32

gray (pytoshop.enums.ColorChannel attribute), 32

grayscale (pytoshop.enums.ColorMode attribute), 33

grayscale (pytoshop.enums.ColorSpace attribute), 34

grayscale\_and\_multichannel\_halftoning\_info (pytoshop.enums.ImageResourceID attribute), 35

grayscale\_and\_multichannel\_transfer\_func (pytoshop.enums.ImageResourceID attribute), 35

green (pytoshop.enums.ChannelId attribute), 32

green (pytoshop.enums.ColorChannel attribute), 32

grid\_and\_guides\_info (pytoshop.enums.ImageResourceID attribute), 35

grid\_hori (pytoshop.image\_resources.GridAndGuidesInfo attribute), 18

grid\_vert (pytoshop.image\_resources.GridAndGuidesInfo attribute), 18

GridAndGuidesInfo (class in pytoshop.image\_resources), 18

Group (class in pytoshop.user.nested\_layers), 9

group\_id (pytoshop.user.nested\_layers.Layer attribute), 10

group\_ids (pytoshop.image\_resources.LayersGroupInfo attribute), 20

GuideDirection (class in pytoshop.enums), 34

GuideResourceBlock (class in pytoshop.image\_resources), 18

guides (pytoshop.image\_resources.GridAndGuidesInfo attribute), 18

## H

hard\_light (pytoshop.enums.BlendMode attribute), 31

hard\_mix (pytoshop.enums.BlendMode attribute), 31

has\_real\_merged\_data (pytoshop.image\_resources.VersionInfo attribute), 21

hdr\_toning\_info (pytoshop.enums.ImageResourceID attribute), 35

Header (class in pytoshop.core), 13

header\_read() (pytoshop.core.Header class method), 14

height (pytoshop.core.Header attribute), 14

height (pytoshop.layers.LayerMask attribute), 24

height (pytoshop.layers.LayerRecord attribute), 26

hks (pytoshop.enums.ColorSpace attribute), 34

horizontal (pytoshop.enums.GuideDirection attribute), 34

hsb (pytoshop.enums.ColorSpace attribute), 34

hue (pytoshop.enums.BlendMode attribute), 31

## I

icc\_profile (pytoshop.enums.ImageResourceID attribute), 35

icc\_untagged\_profile (pytoshop.enums.ImageResourceID attribute), 35

id (pytoshop.tagged\_block.LayerId attribute), 29

id (pytoshop.tagged\_block.LayerNameSource attribute), 29

identifiers (pytoshop.image\_resources.AlphaIdentifiers attribute), 16

Image (class in pytoshop.user.nested\_layers), 9  
 image (pytoshop.layers.ChannelImageData attribute), 21  
 image\_data (pytoshop.core.PsdFile attribute), 14  
 image\_mode\_for\_raw (pytoshop.enums.ImageResourceID attribute), 35  
 image\_ready\_7\_rollover\_expanded\_state (pytoshop.enums.ImageResourceID attribute), 35  
 image\_ready\_data\_sets (pytoshop.enums.ImageResourceID attribute), 35  
 image\_ready\_default\_selected\_state (pytoshop.enums.ImageResourceID attribute), 35  
 image\_ready\_rollover\_expanded\_state (pytoshop.enums.ImageResourceID attribute), 35  
 image\_ready\_save\_layer\_settings (pytoshop.enums.ImageResourceID attribute), 35  
 image\_ready\_variables (pytoshop.enums.ImageResourceID attribute), 35  
 image\_ready\_version (pytoshop.enums.ImageResourceID attribute), 35  
 image\_resources (pytoshop.core.PsdFile attribute), 14  
 ImageData (class in pytoshop.image\_data), 15  
 ImageResourceBlock (class in pytoshop.image\_resources), 18  
 ImageResourceID (class in pytoshop.enums), 34  
 ImageResources (class in pytoshop.image\_resources), 19  
 ImageResourceUnicodeString (class in pytoshop.image\_resources), 19  
 inches (pytoshop.enums.Units attribute), 38  
 indexed (pytoshop.enums.ColorMode attribute), 33  
 indexed\_color\_table\_count (pytoshop.enums.ImageResourceID attribute), 35  
 initial\_fill\_rule\_record (pytoshop.enums.PathRecordType attribute), 37  
 InitialFillRuleRecord (class in pytoshop.path), 27  
 interpolate (pytoshop.image\_resources.PrintFlags attribute), 20  
 invert (pytoshop.tagged\_block.VectorMask attribute), 30  
 invert\_layer\_mask\_when\_blending (pytoshop.layers.LayerMask attribute), 24  
 iptc\_naa (pytoshop.enums.ImageResourceID attribute), 35  
 is\_long\_length() (pytoshop.tagged\_block.TaggedBlock static method), 29

## J

jpeg\_quality (pytoshop.enums.ImageResourceID attribute), 36  
 jump\_to\_xpep (pytoshop.enums.ImageResourceID attribute), 36

## K

kind (pytoshop.layers.GlobalLayerMaskInfo attribute), 22

## L

L (pytoshop.enums.ChannelId attribute), 32  
 L (pytoshop.enums.ColorChannel attribute), 32  
 lab (pytoshop.enums.ColorMode attribute), 33  
 lab (pytoshop.enums.ColorSpace attribute), 34  
 labels (pytoshop.image\_resources.PrintFlags attribute), 20  
 Layer (class in pytoshop.user.nested\_layers), 10  
 layer\_and\_mask\_info (pytoshop.core.PsdFile attribute), 14  
 layer\_color (pytoshop.user.nested\_layers.Layer attribute), 10  
 layer\_comps (pytoshop.enums.ImageResourceID attribute), 36  
 layer\_groups\_enabled (pytoshop.enums.ImageResourceID attribute), 36  
 layer\_info (pytoshop.layers.LayerAndMaskInfo attribute), 23  
 layer\_mask\_disabled (pytoshop.layers.LayerMask attribute), 24  
 layer\_records (pytoshop.layers.LayerInfo attribute), 23  
 layer\_selection\_ids (pytoshop.enums.ImageResourceID attribute), 36  
 layer\_state\_info (pytoshop.enums.ImageResourceID attribute), 36  
 LayerAndMaskInfo (class in pytoshop.layers), 22  
 LayerColor (class in pytoshop.tagged\_block), 28  
 LayerId (class in pytoshop.tagged\_block), 29  
 LayerInfo (class in pytoshop.layers), 23  
 LayerMask (class in pytoshop.layers), 23  
 LayerMaskKind (class in pytoshop.enums), 37  
 LayerNameSource (class in pytoshop.tagged\_block), 29  
 LayerRecord (class in pytoshop.layers), 25  
 layers (pytoshop.user.nested\_layers.Group attribute), 9  
 layers\_group\_info (pytoshop.enums.ImageResourceID attribute), 36  
 LayersGroupInfo (class in pytoshop.image\_resources), 20  
 left (pytoshop.layers.LayerMask attribute), 24  
 left (pytoshop.layers.LayerRecord attribute), 26  
 left (pytoshop.path.ClipboardRecord attribute), 27  
 left (pytoshop.user.nested\_layers.Image attribute), 10  
 length() (pytoshop.blending\_range.BlendingRangePair method), 12

- length() (pytoshop.blending\_range.BlendingRanges method), 13
  - length() (pytoshop.color\_mode.ColorModeData method), 15
  - length() (pytoshop.image\_resources.ImageResourceBlock method), 18
  - length() (pytoshop.image\_resources.ImageResources method), 19
  - length() (pytoshop.layers.LayerMask method), 24
  - length() (pytoshop.path.PathRecord method), 28
  - length() (pytoshop.path.PathResource method), 28
  - length() (pytoshop.tagged\_block.TaggedBlock method), 29
  - lighten (pytoshop.enums.BlendMode attribute), 31
  - lighter\_color (pytoshop.enums.BlendMode attribute), 31
  - lightroom\_workflow (pytoshop.enums.ImageResourceID attribute), 36
  - linear\_burn (pytoshop.enums.BlendMode attribute), 31
  - linear\_dodge (pytoshop.enums.BlendMode attribute), 31
  - linear\_light (pytoshop.enums.BlendMode attribute), 31
  - location (pytoshop.image\_resources.GuideResourceBlock attribute), 18
  - log() (in module pytoshop.util), 43
  - luminosity (pytoshop.enums.BlendMode attribute), 31
- ## M
- mac\_nsprintinfo (pytoshop.enums.ImageResourceID attribute), 36
  - mac\_page\_format\_info (pytoshop.enums.ImageResourceID attribute), 36
  - mac\_print\_info (pytoshop.enums.ImageResourceID attribute), 36
  - magenta (pytoshop.enums.ChannelId attribute), 32
  - magenta (pytoshop.enums.ColorChannel attribute), 32
  - mask (pytoshop.layers.LayerRecord attribute), 26
  - max\_size\_mapping (pytoshop.core.Header attribute), 14
  - measurement\_scale (pytoshop.enums.ImageResourceID attribute), 36
  - metadata (pytoshop.user.nested\_layers.Layer attribute), 10
  - MetadataSetting (class in pytoshop.tagged\_block), 29
  - multichannel (pytoshop.enums.ColorMode attribute), 33
  - multiply (pytoshop.enums.BlendMode attribute), 31
- ## N
- name (pytoshop.image\_resources.ImageResourceBlock attribute), 18
  - name (pytoshop.layers.LayerRecord attribute), 26
  - name (pytoshop.tagged\_block.UnicodeLayerName attribute), 30
  - name (pytoshop.user.nested\_layers.Layer attribute), 11
  - name\_of\_clipping\_path (pytoshop.enums.ImageResourceID attribute), 36
  - needs\_byteswap() (in module pytoshop.util), 43
  - negative (pytoshop.image\_resources.PrintFlags attribute), 20
  - nested\_layers\_to\_psd() (in module pytoshop.user.nested\_layers), 11
  - NestedSectionDividerSetting (class in pytoshop.tagged\_block), 29
  - normal (pytoshop.enums.BlendMode attribute), 31
  - normalize\_image() (in module pytoshop.codecs), 42
  - not\_link (pytoshop.tagged\_block.VectorMask attribute), 30
  - num\_channels (pytoshop.core.Header attribute), 14
- ## O
- onion\_skins (pytoshop.enums.ImageResourceID attribute), 36
  - opacity (pytoshop.layers.GlobalLayerMaskInfo attribute), 22
  - opacity (pytoshop.layers.LayerRecord attribute), 26
  - opacity (pytoshop.user.nested\_layers.Layer attribute), 11
  - open (pytoshop.enums.SectionDividerSetting attribute), 38
  - open\_subpath\_bezier\_knot\_linked (pytoshop.enums.PathRecordType attribute), 37
  - open\_subpath\_bezier\_knot\_unlinked (pytoshop.enums.PathRecordType attribute), 37
  - open\_subpath\_length (pytoshop.enums.PathRecordType attribute), 37
  - OpenSubpathBezierKnotLinked (class in pytoshop.path), 27
  - OpenSubpathBezierKnotUnlinked (class in pytoshop.path), 27
  - OpenSubpathLengthRecord (class in pytoshop.path), 28
  - origin\_path\_info (pytoshop.enums.ImageResourceID attribute), 36
  - overlay (pytoshop.enums.BlendMode attribute), 31
  - overlay\_color\_space (pytoshop.layers.GlobalLayerMaskInfo attribute), 22
- ## P
- pack\_bitflags() (in module pytoshop.util), 43
  - pad() (in module pytoshop.util), 43
  - pantone (pytoshop.enums.ColorSpace attribute), 34
  - pascal\_string\_length() (in module pytoshop.util), 43
  - pass\_through (pytoshop.enums.BlendMode attribute), 31
  - path\_fill\_rule\_record (pytoshop.enums.PathRecordType attribute), 37
  - path\_records (pytoshop.path.PathResource attribute), 28
  - path\_resource (pytoshop.tagged\_block.VectorMask attribute), 30

path\_selection\_state (pytoshop.enums.ImageResourceID attribute), 36

PathFillRuleRecord (class in pytoshop.path), 28

PathRecord (class in pytoshop.path), 28

PathRecordType (class in pytoshop.enums), 37

PathResource (class in pytoshop.path), 28

picas (pytoshop.enums.Units attribute), 38

pin\_light (pytoshop.enums.BlendMode attribute), 31

pixel\_aspect\_ratio (pytoshop.enums.ImageResourceID attribute), 36

pixel\_data\_irrelevant (pytoshop.layers.LayerRecord attribute), 26

points (pytoshop.enums.Units attribute), 38

position\_relative\_to\_layer (pytoshop.layers.LayerMask attribute), 24

pprint\_layers() (in module pytoshop.user.nested\_layers), 11

print\_flags (pytoshop.enums.ImageResourceID attribute), 36

print\_flags (pytoshop.image\_resources.PrintFlags attribute), 20

print\_flags\_info (pytoshop.enums.ImageResourceID attribute), 36

print\_info (pytoshop.enums.ImageResourceID attribute), 36

print\_info\_cs5 (pytoshop.enums.ImageResourceID attribute), 36

print\_scale (pytoshop.enums.ImageResourceID attribute), 36

print\_style (pytoshop.enums.ImageResourceID attribute), 36

PrintFlags (class in pytoshop.image\_resources), 20

PrintScale (class in pytoshop.image\_resources), 20

PrintScaleStyle (class in pytoshop.enums), 37

psb (pytoshop.enums.Version attribute), 38

psd (pytoshop.enums.Version attribute), 38

psd\_to\_nested\_layers() (in module pytoshop.user.nested\_layers), 11

PsdFile (class in pytoshop.core), 14

pytoshop.blending\_range (module), 12

pytoshop.codecs (module), 38

pytoshop.color\_mode (module), 15

pytoshop.core (module), 13

pytoshop.enums (module), 31

pytoshop.image\_data (module), 15

pytoshop.image\_resources (module), 16

pytoshop.layers (module), 21

pytoshop.path (module), 27

pytoshop.tagged\_block (module), 28

pytoshop.user.nested\_layers (module), 9

pytoshop.util (module), 42

## Q

quick\_mask\_info (pytoshop.enums.ImageResourceID at-

tribute), 36

## R

raw (pytoshop.enums.Compression attribute), 34

read() (in module pytoshop), 9

read() (pytoshop.blending\_range.BlendingRange class method), 12

read() (pytoshop.blending\_range.BlendingRangePair class method), 12

read() (pytoshop.blending\_range.BlendingRanges class method), 13

read() (pytoshop.color\_mode.ColorModeData class method), 15

read() (pytoshop.core.PsdFile class method), 14

read() (pytoshop.image\_data.ImageData class method), 15

read() (pytoshop.image\_resources.GuideResourceBlock class method), 18

read() (pytoshop.image\_resources.ImageResourceBlock class method), 18

read() (pytoshop.image\_resources.ImageResources class method), 19

read() (pytoshop.layers.ChannelImageData class method), 22

read() (pytoshop.layers.GlobalLayerMaskInfo class method), 22

read() (pytoshop.layers.LayerAndMaskInfo class method), 23

read() (pytoshop.layers.LayerInfo class method), 23

read() (pytoshop.layers.LayerMask class method), 24

read() (pytoshop.layers.LayerRecord class method), 26

read() (pytoshop.path.PathRecord class method), 28

read() (pytoshop.path.PathResource class method), 28

read() (pytoshop.tagged\_block.TaggedBlock class method), 29

read\_channel\_data() (pytoshop.layers.LayerRecord method), 26

read\_data() (pytoshop.image\_resources.AlphaIdentifiers class method), 16

read\_data() (pytoshop.image\_resources.BackgroundColor class method), 16

read\_data() (pytoshop.image\_resources.BorderInfo class method), 16

read\_data() (pytoshop.image\_resources.CopyrightFlag class method), 17

read\_data() (pytoshop.image\_resources.DocumentSpecificIdsSeedNumber class method), 17

read\_data() (pytoshop.image\_resources.EffectsVisible class method), 17

read\_data() (pytoshop.image\_resources.GenericImageResourceBlock class method), 17

read\_data() (pytoshop.image\_resources.GlobalAltitude class method), 17

read\_data() (pytoshop.image\_resources.GlobalAngle class method), 18  
 read\_data() (pytoshop.image\_resources.GridAndGuidesInfo class method), 18  
 read\_data() (pytoshop.image\_resources.ImageResourceBlock class method), 19  
 read\_data() (pytoshop.image\_resources.ImageResourceUnicodeString class method), 19  
 read\_data() (pytoshop.image\_resources.LayersGroupInfo class method), 20  
 read\_data() (pytoshop.image\_resources.PrintFlags class method), 20  
 read\_data() (pytoshop.image\_resources.PrintScale class method), 20  
 read\_data() (pytoshop.image\_resources.Url class method), 21  
 read\_data() (pytoshop.image\_resources.VersionInfo class method), 21  
 read\_data() (pytoshop.path.ClipboardRecord class method), 27  
 read\_data() (pytoshop.path.InitialFillRuleRecord class method), 27  
 read\_data() (pytoshop.path.PathFillRuleRecord class method), 28  
 read\_data() (pytoshop.path.PathRecord class method), 28  
 read\_data() (pytoshop.tagged\_block.GenericTaggedBlock class method), 28  
 read\_data() (pytoshop.tagged\_block.LayerColor class method), 29  
 read\_data() (pytoshop.tagged\_block.LayerId class method), 29  
 read\_data() (pytoshop.tagged\_block.LayerNameSource class method), 29  
 read\_data() (pytoshop.tagged\_block.MetadataSetting class method), 29  
 read\_data() (pytoshop.tagged\_block.TaggedBlock class method), 30  
 read\_data() (pytoshop.tagged\_block.UnicodeLayerName class method), 30  
 read\_data() (pytoshop.tagged\_block.VectorMask class method), 30  
 read\_pascal\_string() (in module pytoshop.util), 43  
 read\_unicode\_string() (in module pytoshop.util), 43  
 read\_value() (in module pytoshop.util), 43  
 reader (pytoshop.image\_resources.VersionInfo attribute), 21  
 real\_bottom (pytoshop.layers.LayerMask attribute), 24  
 real\_flags (pytoshop.layers.LayerMask attribute), 24  
 real\_height (pytoshop.layers.LayerMask attribute), 24  
 real\_left (pytoshop.layers.LayerMask attribute), 24  
 real\_right (pytoshop.layers.LayerMask attribute), 24  
 real\_shape (pytoshop.layers.LayerMask attribute), 24  
 real\_top (pytoshop.layers.LayerMask attribute), 24  
 real\_user\_layer\_mask (pytoshop.enums.ChannelId attribute), 32  
 real\_user\_layer\_mask (pytoshop.enums.ColorChannel attribute), 32  
 real\_user\_mask\_background (pytoshop.layers.LayerMask attribute), 24  
 real\_width (pytoshop.layers.LayerMask attribute), 24  
 red (pytoshop.enums.ChannelId attribute), 32  
 red (pytoshop.enums.ColorChannel attribute), 32  
 registration\_marks (pytoshop.image\_resources.PrintFlags attribute), 20  
 resolution (pytoshop.path.ClipboardRecord attribute), 27  
 resolution\_info (pytoshop.enums.ImageResourceID attribute), 36  
 resource\_id (pytoshop.image\_resources.GenericImageResourceBlock attribute), 17  
 resource\_id (pytoshop.image\_resources.ImageResourceBlock attribute), 19  
 rgb (pytoshop.enums.ColorMode attribute), 33  
 rgb (pytoshop.enums.ColorSpace attribute), 34  
 right (pytoshop.layers.LayerMask attribute), 24  
 right (pytoshop.layers.LayerRecord attribute), 26  
 right (pytoshop.path.ClipboardRecord attribute), 27  
 right (pytoshop.user.nested\_layers.Image attribute), 10  
 rle (pytoshop.enums.Compression attribute), 34

## S

saturation (pytoshop.enums.BlendMode attribute), 31  
 scale (pytoshop.image\_resources.PrintScale attribute), 20  
 screen (pytoshop.enums.BlendMode attribute), 31  
 SectionDividerSetting (class in pytoshop.enums), 37  
 SectionDividerSetting (class in pytoshop.tagged\_block), 29  
 set\_channel() (in module pytoshop.util), 44  
 set\_channel() (pytoshop.layers.LayerRecord method), 26  
 set\_channel() (pytoshop.user.nested\_layers.Image method), 10  
 shape (pytoshop.core.Header attribute), 14  
 shape (pytoshop.image\_data.ImageData attribute), 16  
 shape (pytoshop.layers.ChannelImageData attribute), 22  
 shape (pytoshop.layers.LayerMask attribute), 24  
 shape (pytoshop.layers.LayerRecord attribute), 26  
 sheet\_disclosure (pytoshop.enums.ImageResourceID attribute), 36  
 size\_to\_fit (pytoshop.enums.PrintScaleStyle attribute), 37  
 slices (pytoshop.enums.ImageResourceID attribute), 36  
 soft\_light (pytoshop.enums.BlendMode attribute), 31  
 spot\_half\_tone (pytoshop.enums.ImageResourceID attribute), 36  
 src (pytoshop.blending\_range.BlendingRangePair attribute), 13  
 style (pytoshop.image\_resources.PrintScale attribute), 20  
 subtract (pytoshop.enums.BlendMode attribute), 31



## T

TaggedBlock (class in pytoshop.tagged\_block), 29  
 thumbnail\_resource (pytoshop.enums.ImageResourceID attribute), 36  
 timeline\_info (pytoshop.enums.ImageResourceID attribute), 36  
 top (pytoshop.layers.LayerMask attribute), 24  
 top (pytoshop.layers.LayerRecord attribute), 26  
 top (pytoshop.path.ClipboardRecord attribute), 27  
 top (pytoshop.user.nested\_layers.Image attribute), 10  
 total\_length() (pytoshop.blending\_range.BlendingRangePair method), 13  
 total\_length() (pytoshop.blending\_range.BlendingRanges method), 13  
 total\_length() (pytoshop.image\_resources.ImageResourceBlock method), 19  
 total\_length() (pytoshop.image\_resources.ImageResources method), 19  
 total\_length() (pytoshop.layers.LayerMask method), 24  
 total\_length() (pytoshop.tagged\_block.TaggedBlock method), 30  
 toyo\_88\_colorfinder\_1050 (pytoshop.enums.ColorSpace attribute), 34  
 trace\_read() (in module pytoshop.util), 44  
 trace\_write() (in module pytoshop.util), 44  
 transparency (pytoshop.enums.ChannelId attribute), 32  
 transparency (pytoshop.enums.ColorChannel attribute), 32  
 transparency\_index (pytoshop.enums.ImageResourceID attribute), 36  
 transparency\_protected (pytoshop.layers.LayerRecord attribute), 26  
 trumatch (pytoshop.enums.ColorSpace attribute), 34  
 type (pytoshop.path.PathRecord attribute), 28

## U

unicode\_alpha\_names (pytoshop.enums.ImageResourceID attribute), 36  
 unicode\_string\_length() (in module pytoshop.util), 44  
 UnicodeAlphaNames (class in pytoshop.image\_resources), 21  
 UnicodeLayerName (class in pytoshop.tagged\_block), 30  
 unit (pytoshop.image\_resources.BorderInfo attribute), 16  
 Units (class in pytoshop.enums), 38  
 unpack\_bitflags() (in module pytoshop.util), 44  
 Url (class in pytoshop.image\_resources), 21  
 url (pytoshop.enums.ImageResourceID attribute), 36  
 url (pytoshop.image\_resources.Url attribute), 21  
 url\_list (pytoshop.enums.ImageResourceID attribute), 36  
 use\_alpha\_channel (pytoshop.layers.LayerInfo attribute), 23  
 use\_value\_stored\_per\_layer (pytoshop.enums.LayerMaskKind attribute),

37  
 user\_defined (pytoshop.enums.PrintScaleStyle attribute), 37  
 user\_layer\_mask (pytoshop.enums.ChannelId attribute), 32  
 user\_layer\_mask (pytoshop.enums.ColorChannel attribute), 32  
 user\_mask\_density (pytoshop.layers.LayerMask attribute), 25  
 user\_mask\_feather (pytoshop.layers.LayerMask attribute), 25  
 user\_mask\_from\_rendering\_other\_data (pytoshop.layers.LayerMask attribute), 25

## V

value (pytoshop.image\_resources.ImageResourceUnicodeString attribute), 19  
 vector\_mask\_density (pytoshop.layers.LayerMask attribute), 25  
 vector\_mask\_feather (pytoshop.layers.LayerMask attribute), 25  
 VectorMask (class in pytoshop.tagged\_block), 30  
 Version (class in pytoshop.enums), 38  
 version (pytoshop.core.Header attribute), 14  
 version (pytoshop.image\_resources.GridAndGuidesInfo attribute), 18  
 version (pytoshop.image\_resources.VersionInfo attribute), 21  
 version (pytoshop.tagged\_block.VectorMask attribute), 30  
 version\_1 (pytoshop.enums.Version attribute), 38  
 version\_2 (pytoshop.enums.Version attribute), 38  
 version\_info (pytoshop.enums.ImageResourceID attribute), 36  
 VersionInfo (class in pytoshop.image\_resources), 21  
 vertical (pytoshop.enums.GuideDirection attribute), 34  
 visible (pytoshop.image\_resources.EffectsVisible attribute), 17  
 visible (pytoshop.layers.LayerRecord attribute), 26  
 visible (pytoshop.user.nested\_layers.Layer attribute), 11  
 vivid\_light (pytoshop.enums.BlendMode attribute), 31

## W

watermark (pytoshop.enums.ImageResourceID attribute), 36  
 white0 (pytoshop.blending\_range.BlendingRange attribute), 12  
 white1 (pytoshop.blending\_range.BlendingRange attribute), 12  
 width (pytoshop.core.Header attribute), 14  
 width (pytoshop.layers.LayerMask attribute), 25  
 width (pytoshop.layers.LayerRecord attribute), 26  
 win\_devmode (pytoshop.enums.ImageResourceID attribute), 37

- workflow\_url (pytoshop.enums.ImageResourceID attribute), 37
  - WorkflowUrl (class in pytoshop.image\_resources), 21
  - write() (pytoshop.blending\_range.BlendingRange method), 12
  - write() (pytoshop.blending\_range.BlendingRangePair method), 13
  - write() (pytoshop.blending\_range.BlendingRanges method), 13
  - write() (pytoshop.color\_mode.ColorModeData method), 15
  - write() (pytoshop.core.Header method), 14
  - write() (pytoshop.core.PsdFile method), 14
  - write() (pytoshop.image\_data.ImageData method), 16
  - write() (pytoshop.image\_resources.GuideResourceBlock method), 18
  - write() (pytoshop.image\_resources.ImageResourceBlock method), 19
  - write() (pytoshop.image\_resources.ImageResources method), 19
  - write() (pytoshop.layers.ChannelImageData method), 22
  - write() (pytoshop.layers.GlobalLayerMaskInfo method), 22
  - write() (pytoshop.layers.LayerAndMaskInfo method), 23
  - write() (pytoshop.layers.LayerInfo method), 23
  - write() (pytoshop.layers.LayerMask method), 25
  - write() (pytoshop.layers.LayerRecord method), 26
  - write() (pytoshop.path.PathRecord method), 28
  - write() (pytoshop.path.PathResource method), 28
  - write() (pytoshop.tagged\_block.TaggedBlock method), 30
  - write\_channel\_data() (pytoshop.layers.LayerRecord method), 27
  - write\_data() (pytoshop.image\_resources.AlphaIdentifiers method), 16
  - write\_data() (pytoshop.image\_resources.BackgroundColor method), 16
  - write\_data() (pytoshop.image\_resources.BorderInfo method), 16
  - write\_data() (pytoshop.image\_resources.CopyrightFlag method), 17
  - write\_data() (pytoshop.image\_resources.DocumentSpecificIds method), 17
  - write\_data() (pytoshop.image\_resources.EffectsVisible method), 17
  - write\_data() (pytoshop.image\_resources.GenericImageResourceBlock method), 17
  - write\_data() (pytoshop.image\_resources.GlobalAltitude method), 17
  - write\_data() (pytoshop.image\_resources.GlobalAngle method), 18
  - write\_data() (pytoshop.image\_resources.GridAndGuidesInfo method), 18
  - write\_data() (pytoshop.image\_resources.ImageResourceBlock method), 19
  - write\_data() (pytoshop.image\_resources.ImageResourceUnicodeString method), 19
  - write\_data() (pytoshop.image\_resources.LayersGroupInfo method), 20
  - write\_data() (pytoshop.image\_resources.PrintFlags method), 20
  - write\_data() (pytoshop.image\_resources.PrintScale method), 20
  - write\_data() (pytoshop.image\_resources.Url method), 21
  - write\_data() (pytoshop.image\_resources.VersionInfo method), 21
  - write\_data() (pytoshop.path.ClipboardRecord method), 27
  - write\_data() (pytoshop.path.InitialFillRuleRecord method), 27
  - write\_data() (pytoshop.path.PathFillRuleRecord method), 28
  - write\_data() (pytoshop.path.PathRecord method), 28
  - write\_data() (pytoshop.tagged\_block.GenericTaggedBlock method), 28
  - write\_data() (pytoshop.tagged\_block.LayerColor method), 29
  - write\_data() (pytoshop.tagged\_block.LayerId method), 29
  - write\_data() (pytoshop.tagged\_block.LayerNameSource method), 29
  - write\_data() (pytoshop.tagged\_block.MetadataSetting method), 29
  - write\_data() (pytoshop.tagged\_block.TaggedBlock method), 30
  - write\_data() (pytoshop.tagged\_block.UnicodeLayerName method), 30
  - write\_data() (pytoshop.tagged\_block.VectorMask method), 30
  - write\_pascal\_string() (in module pytoshop.util), 44
  - write\_unicode\_string() (in module pytoshop.util), 44
  - write\_value() (in module pytoshop.util), 44
  - writer (pytoshop.image\_resources.VersionInfo attribute), 21
- X**
- xSeedNumber (pytoshop.image\_resources.PrintScale attribute), 20
  - xmp\_metadata (pytoshop.enums.ImageResourceID attribute), 37
- Y**
- y (pytoshop.image\_resources.PrintScale attribute), 21
  - yellow (pytoshop.enums.ChannelId attribute), 32
  - yellow (pytoshop.enums.ColorChannel attribute), 32
- Z**
- zip (pytoshop.enums.Compression attribute), 34

zip\_prediction (pytoshop.enums.Compression attribute),  
34