OpenSlide

OpenSlide Python is a Python interface to the OpenSlide library.

OpenSlide is a C library that provides a simple interface for reading whole-slide images, also known as virtual slides, which are high-resolution images used in digital pathology. These images can occupy tens of gigabytes when uncompressed, and so cannot be easily read using standard tools or libraries, which are designed for images that can be comfortably uncompressed into RAM. Whole-slide images are typically multi-resolution; OpenSlide allows reading a small amount of image data at the resolution closest to a desired zoom level.

OpenSlide can read virtual slides in several formats:

- Aperio (`.svs`, `.tif`)
- DICOM (`.dcm`)
- Hamamatsu (`.ndpi`, `.vms`, `.vmu`)
- Leica (`.scn`)
- MIRAX (`.mrxs`)
- Philips (`.tiff`)
- Sakura (`.svslide`)
- Trestle (`.tif`)
- Ventana (`.bif`, `.tif`)
- Zeiss (`.czi`)

- Generic tiled TIFF (`.tif`)

Installing

OpenSlide Python requires OpenSlide, which must be installed separately. If you intend to use OpenSlide only with Python, the easiest way to get it is to install the openslide-bin Python package with `pip install openslide-bin`.

On Linux and macOS, you can also install both OpenSlide and OpenSlide Python with a package manager that packages both, such as Anaconda, DNF or Apt on Linux systems, or MacPorts on macOS systems. Or, you can install OpenSlide Python with pip after installing OpenSlide with a package manager or from source. Except for pip, do not mix OpenSlide and OpenSlide Python from different package managers (for example, OpenSlide from MacPorts and OpenSlide Python from Anaconda), since you'll get library conflicts.

On Windows, you can also download the OpenSlide Windows binaries and extract them to a known path. Then, import `openslide` inside a `with os.add_dll_directory()` statement:

```
# The path can also be read from a config file, etc.
OPENSLIDE_PATH = r'c:\path\to\openslide-win64\bin'
import os
if hasattr(os, 'add_dll_directory'):
  # Windows
  with os.add_dll_directory(OPENSLIDE_PATH):
    import openslide
else:
  import openslide
```

Basic usage

OpenSlide objects

class openslide.**OpenSlide**(*filename: str | bytes | PathLike[Any]*)

An open whole-slide image.

If any operation on the object fails, **OpenSlideError** is raised. OpenSlide has latching error semantics: once **OpenSlideError** is raised, all future operations on the **OpenSlide**, other than **close()**, will also raise **OpenSlideError**.

close() is called automatically when the object is deleted. The object may be used as a context manager, in which case it will be closed upon exiting the context.

Parameters:

filename – the file to open

Raises:

- **OpenSlideUnsupportedFormatError** – if the file is not recognized by OpenSlide
- **OpenSlideError** – if the file is recognized but an error occurred

classmethod **detect_format**(*filename: str | bytes | PathLike[Any]*) → str | None

Return a string describing the format vendor of the specified file. This string is also accessible via the **PROPERTY_NAME_VENDOR** property.

If the file is not recognized, return **None**.

Parameters:

filename – the file to examine

level_count

The number of levels in the slide. Levels are numbered from `0` (highest resolution) to `level_count - 1` (lowest resolution).

Type:

int

dimensions

A `(width, height)` tuple for level 0 of the slide.

Type:

tuple[int, int]

level_dimensions

A tuple of `(width, height)` tuples, one for each level of the slide. `level_dimensions[k]` are the dimensions of level `k`.

Type:

tuple[tuple[int, int], ...]

level_downsamples

A tuple of downsample factors for each level of the slide. `level_downsamples[k]` is the downsample factor of level `k`.

Type:

tuple[float, ...]

properties

Metadata about the slide, in the form of a **Mapping** from OpenSlide property name to property value. OpenSlide provides some Standard properties, plus additional properties that vary by slide format.

Type:

Mapping[str, str]

associated_images

Images, such as label or macro images, which are associated with this slide. This is a **Mapping** from image name to RGBA **Image**.

Unlike in the C interface, these images are not premultiplied.

Type:

Mapping[str, *Image*]

color_profile

The embedded color profile for this slide, or **None** if not available.

Type:

ImageCmsProfile | None

read_region(*location: tuple[int, int], level: int, size: tuple[int, int]*) → Image

Return an RGBA **Image** containing the contents of the specified region.

Unlike in the C interface, the image data is not premultiplied.

Parameters:

- **location** – `(x, y)` tuple giving the top left pixel in the level 0 reference frame
- **level** – the level number
- **size** – `(width, height)` tuple giving the region size

get_best_level_for_downsample(*downsample: float*) → int

Return the best level for displaying the given downsample.

Parameters:

downsample – the desired downsample factor

get_thumbnail(*size: tuple[int, int]*) → Image

Return an **Image** containing an RGB thumbnail of the slide.

Parameters:

size – the maximum size of the thumbnail as a `(width, height)` tuple

set_cache(*cache: OpenSlideCache*) → None

Use the specified **OpenSlideCache** to store recently decoded slide tiles. By default, the **OpenSlide** has a private cache with a default size.

Parameters:

cache – a cache object

Raises:

OpenSlideVersionError – if OpenSlide is older than version 4.0.0

close() → None

Close the OpenSlide object.

Color management

Every slide region, associated image, thumbnail, and Deep Zoom tile produced by OpenSlide Python includes a reference to an ICC color profile whenever a profile is available for the underlying pixel

data. Profiles are stored as a **bytes** object in **Image.info['icc_profile']**. If no profile is available, the **icc_profile** dictionary key is absent.

To include the profile in an image file when saving the image to disk:

```
image.save(filename, icc_profile=image.info.get('icc_profile'))
```

To perform color conversions using the profile, import it into **ImageCms**. For example, to synthesize an sRGB profile and use it to transform an image for display, with the default rendering intent of the image's profile:

```
from io import BytesIO
from PIL import ImageCms

fromProfile = ImageCms.getOpenProfile(BytesIO(image.info['icc_profile']))
toProfile = ImageCms.createProfile('sRGB')
intent = ImageCms.getDefaultIntent(fromProfile)
ImageCms.profileToProfile(
  image, fromProfile, toProfile, intent, 'RGBA', True, 0
)
```

When converting Deep Zoom tiles, use `'RGB'` instead of `'RGBA'`.

All pyramid regions in a slide have the same profile, but each associated image can have its own profile. As a convenience, the former is also available as **OpenSlide.color_profile**, already parsed into an **ImageCmsProfile** object. You can save processing time by building an **ImageCmsTransform** for the slide and reusing it for multiple slide regions:

```
toProfile = ImageCms.createProfile('sRGB')
intent = ImageCms.getDefaultIntent(slide.color_profile)
transform = ImageCms.buildTransform(
  slide.color_profile, toProfile, 'RGBA', 'RGBA', intent, 0
)
# for each region image:
ImageCms.applyTransform(image, transform, True)
```

Caching

class openslide.**OpenSlideCache**(*capacity: int*)

An in-memory tile cache.

Tile caches can be attached to one or more **OpenSlide** objects with **OpenSlide.set_cache()** to cache recently-decoded tiles. By default, each **OpenSlide** has its own cache with a default size.

Parameters:

capacity – the cache capacity in bytes

Raises:

OpenSlideVersionError – if OpenSlide is older than version 4.0.0

Standard properties

The **openslide** module provides attributes containing the names of some commonly-used OpenSlide properties.

openslide.**PROPERTY_NAME_COMMENT**

The name of the property containing a slide's comment, if any.

openslide.**PROPERTY_NAME_VENDOR**

The name of the property containing an identification of the vendor.

openslide.**PROPERTY_NAME_QUICKHASH1**

The name of the property containing the "quickhash-1" sum.

openslide.**PROPERTY_NAME_BACKGROUND_COLOR**

The name of the property containing a slide's background color, if any. It is represented as an RGB hex triplet.

openslide.**PROPERTY_NAME_OBJECTIVE_POWER**

The name of the property containing a slide's objective power, if known.

openslide.**PROPERTY_NAME_MPP_X**

The name of the property containing the number of microns per pixel in the X dimension of level 0, if known.

openslide.**PROPERTY_NAME_MPP_Y**

The name of the property containing the number of microns per pixel in the Y dimension of level 0, if known.

openslide.**PROPERTY_NAME_BOUNDS_X**

The name of the property containing the X coordinate of the rectangle bounding the non-empty region of the slide, if available.

openslide.**PROPERTY_NAME_BOUNDS_Y**

The name of the property containing the Y coordinate of the rectangle bounding the non-empty region of the slide, if available.

openslide.**PROPERTY_NAME_BOUNDS_WIDTH**

The name of the property containing the width of the rectangle bounding the non-empty region of the slide, if available.

openslide.**PROPERTY_NAME_BOUNDS_HEIGHT**

The name of the property containing the height of the rectangle bounding the non-empty region of the slide, if available.

Exceptions

exception openslide.**OpenSlideError**

An error produced by the OpenSlide library.

Once **OpenSlideError** has been raised by a particular **OpenSlide**, all future operations on that **OpenSlide** (other than **close()**) will also raise **OpenSlideError**.

exception openslide.**OpenSlideUnsupportedFormatError**

OpenSlide does not support the requested file. Subclass of **OpenSlideError**.

exception openslide.**OpenSlideVersionError**

This version of OpenSlide does not support the requested functionality. Subclass of **OpenSlideError**.

Wrapping a Pillow Image

class openslide.**AbstractSlide**

The abstract base class of **OpenSlide** and **ImageSlide**.

class openslide.**ImageSlide**(*file: str | bytes | PathLike[Any] | Image*)

A wrapper around an **Image** object that provides an **OpenSlide**-compatible API.

Parameters:

file – a filename or **Image** object

Raises:

OSError – if the file cannot be opened

openslide.**open_slide**(*filename: str | bytes | PathLike[Any]*) → OpenSlide | ImageSlide

Return an **OpenSlide** for whole-slide images and an **ImageSlide** for other types of images.

Parameters:

filename – the file to open

Raises:

- **OpenSlideError** – if the file is recognized by OpenSlide but an error occurred
- **OSError** – if the file is not recognized at all

Deep Zoom support

OpenSlide Python provides functionality for generating individual Deep Zoom tiles from slide objects. This is useful for displaying whole-slide images in a web browser without converting the entire slide to Deep Zoom or a similar format.

class openslide.deepzoom.**DeepZoomGenerator**(*osr: AbstractSlide*, *tile_size: int = 254*, *overlap: int = 1*, *limit_bounds: bool = False*)

A Deep Zoom generator that wraps an **OpenSlide** object, **ImageSlide** object, or user-provided instance of **AbstractSlide**.

Parameters:

- **osr** – the slide object
- **tile_size** – the width and height of a single tile. For best viewer performance, `tile_size + 2 * overlap` should be a power of two.
- **overlap** – the number of extra pixels to add to each interior edge of a tile
- **limit_bounds** – **True** to render only the non-empty slide region

level_count

The number of Deep Zoom levels in the image.

Type:

int

tile_count

The total number of Deep Zoom tiles in the image.

Type:

int

level_tiles

A tuple of `(tiles_x, tiles_y)` tuples for each Deep Zoom level. `level_tiles[k]` are the tile counts of level `k`.

Type:

tuple[tuple[int, int], …]

level_dimensions

A tuple of `(pixels_x, pixels_y)` tuples for each Deep Zoom level. `level_dimensions[k]` are the dimensions of level `k`.

Type:

tuple[tuple[int, int], …]

get_dzi(*format: str*) → str

Return a string containing the XML metadata for the Deep Zoom `.dzi` file.

Parameters:

format – the delivery format of the individual tiles (`png` or `jpeg`)

get_tile(*level: int*, *address: tuple[int, int]*) → Image

Return an RGB **Image** for a tile.

Parameters:

- **level** – the Deep Zoom level
- **address** – the address of the tile within the level as a `(column, row)` tuple

get_tile_coordinates(*level: int*, *address: tuple[int, int]*) → tuple[tuple[int, int], int, tuple[int, int]]

Return the **OpenSlide.read_region()** arguments corresponding to the specified tile.

Most applications should use **get_tile()** instead.

Parameters:

- **level** – the Deep Zoom level
- **address** – the address of the tile within the level as a `(column, row)` tuple

get_tile_dimensions(*level: int*, *address: tuple[int, int]*) → tuple[int, int]

Return a `(pixels_x, pixels_y)` tuple for the specified tile.

Parameters:

- **level** – the Deep Zoom level
- **address** – the address of the tile within the level as a `(column, row)` tuple