vignette Documentation

Release 4.4.0

Hg

May 07, 2018

Contents

Generate and retrieve thumbnails according to the Freedesktop.org thumbnail standard.

Thumbnails are stored in a shared directory so other apps following the standard can reuse them without having to generate their own thumbnails.

vignette can typically be used in file managers, image browsers, etc.

Thumbnails are not limited to image files on disk but can be managed for other file types, for example videos or documents but also for any URL, for example a web browser could store thumbnails for recently visited pages or bookmarks.

vignette by itself can only generate thumbnails for local image files but can retrieve thumbnail for any file or URL, if another app generated a thumbnail for it. An app can also generate a thumbnail by its own means and use *vignette* to push that thumbnail to the store.

vignette has optional support for extra backends like `ffmpegthumbnailer`, `poppler-utils`, `ooo-thumbnailer`, if these tools are installed.

CHAPTER 1

Summary of the FreeDesktop standard

- Thumbnails can be generated for any file or URL (should it be an image, a video, a webpage)
- Thumbnails are be stored in `~/.cache/thumbnails` in PNG format
- The store can contain 2 sizes for thumbnails: 128x128 and 256x256, stored respectively in `~/.cache/thumbnails/normal` and `~/.cache/thumbnails/large`
- Files or URLs thumbnailed must have a "last modified time" (`mtime` for short) to detect obsolescence of thumbnails
- Additional metadata can be put in thumbnails, as key/value pairs, in the PNG text fields, such as the duration for a video file
- There are 2 attributes required to be put in the metadata: the source URI and the mtime, to ensure validity of thumbnails
- To avoid useless retries, if a thumbnail can't be generated by an app (e.g. because of an erroneous file or an unsupported format), a "fail-file" can be written in `~/.cache/thumbnails/fail/<appname-version>`
- Having a failed thumbnail doesn't mean another app cannot succeed (for example because of format support), then the successful thumbnail can be used everywhere

For more details, read the Freedesktop.org thumbnail standard.

CHAPTER 2

Module functions

2.1 Querying

These functions are used to find if a thumbnail exists or find info about a thumbnail. They do not generate thumbnails or "fail-files", and can be used with files or URLs, that can refer to non-images:

- *try_get_thumbnail*
- *build_thumbnail_path*
- *is_thumbnail_failed*

These functions generally file paths or URLs as `src` argument. If it is an URL, the `mtime` argument must be specified, because *vignette* can only determine the mtime of local files.

2.2 Generating

These functions do generate thumbnails. They require their `src` argument to be a local file and the file must be an image. For instance, they do not handle videos and cannot generate thumbnails for them.

If they cannot generate a thumbnail for a file (for example because the format is not an image or because the file is corrupt), they may create a fail-file.

- *get_thumbnail*
- *create_thumbnail*

2.3 Storing

vignette can only generate thumbnails for image files, not other formats like videos or webpages.

However, if the application using *vignette* can generate thumbnails for other file types or URLs by its own means, it can still use *vignette* to put the thumbnails in the store, and be able to retrieve them afterwards with *try_get_thumbnail*, or to let other apps find them.

- *put_thumbnail*
- *put_fail*

These functions generally file paths or URLs as `src` argument. If it is an URL, the `mtime` argument must be specified, because *vignette* can only determine the mtime of local files.

CHAPTER 3

General notes

3.1 MTime

The last-modification time of a source file to create a thumbnail for is very important in the standard, since it allows to easily determine if a thumbnail is still relevant.

A thumbnail is identified by a source URL and the last-modification time of the source (*mtime* for short). If a thumbnails exists for the source URL but the source's *mtime* is different, the thumbnail is considered obsolete and simply ignored.

Many functions of *vignette* take a `mtime` argument that is optional if the `src` argument refers to a local file, but mandatory if it is an URL.

CHAPTER 4

Examples

Just query thumbnail of a local image, automatically creating it if necessary:

```
import vignette

thumb_image = vignette.get_thumbnail('/my/file.jpg')
local_app_display(thumb_image)
```

Ask for a thumbnail or generate it manually, for example a web-browser generating pages previews, that this module can't do itself:

```
import vignette

orig_url = 'http://example.com/file.pdf'
thumb_image = vignette.try_get_thumbnail(orig_url, mtime=0) # mtime is not used in␣
↪this example

if not thumb_image:
  thumb_image = vignette.create_temp('large')
  try:
    local_app_make_preview(orig_url, thumb_image)
  except NetworkError:
    vignette.put_fail(orig_url, 'mybrowser-1.0', mtime=0)
  else:
    thumb_image = vignette.put_thumbnail(orig_url, 'large', thumb_image, mtime=0)
  if vignette.is_thumbnail_failed(orig_url, 'mybrowser-1.0'):
    thumb_image = 'error.png'

local_app_display(thumb_image)
```

CHAPTER 5

Module contents

vignette.**get_thumbnail**(*src*, *size=None*, *use_fail_appname=None*)

Get the path of the thumbnail and create it if necessary.

If a thumbnail exists and is valid, return it.

If the thumbnail cannot be found, and a previous failure info file had been created with the given app name, the thumbnail generation is not attempted and None is returned.

Else, thumbnail generation is done. If an error occurs during generation, the function returns None. If *use_fail_appname* is specified, a fail-file is generated in case of error.

Parameters

- **src** (*str*) – path of the source file. Must be an image file. Cannot be a URL.
- **size** – desired size of thumbnail. Can be any of 'large', 256 for large thumbnails or 'normal', 128 for small thumbnails. If None, searches for any size.
- **moreinfo** (*dict*) – additional optional key/values to store in the thumbnail file. Used only if a thumbnail is generated.
- **use_fail_appname** (*str*) – app name to use when creating a failure info.

Returns the path of the thumbnail, or None if it couldn't be generated

Return type str

vignette.**try_get_thumbnail**(*src*, *size=None*, *mtime=None*)

Get the path of the thumbnail or None if it doesn't exist.

If a thumbnail exists but is obsolete (different mtime), None is returned.

Parameters

- **src** (*str*) – path or URI of the source file.
- **size** – desired size of thumbnail. Can be 'large', 256 or 'normal', 128. If None, tries with the large thumbnail size first, then with the normal size.
- **mtime** (*int*) – mtime of the source file. Optional only if *src* is a local file.

Returns path of the thumbnail if it exists and is valid, else None

Return type str

vignette.**build_thumbnail_path**(*src*, *size*)
Get the path of the potential thumbnail.

The thumbnail file may or may not exist. Even if it exists, the thumbnail may be obsolete. Use *try_get_thumbnail* if needing to check if the thumbnail exist and is valid.

Parameters

- **src** (*str*) – path or URI of the source file.
- **size** – desired size of thumbnail. Can be any of 'large', 256 for large thumbnails or 'normal', 128 for small thumbnails.

Returns path of where the thumbnail should be

Return type str

vignette.**create_thumbnail**(*src*, *size*, *moreinfo=None*, *use_fail_appname=None*)
Generate a thumbnail for *src*, even if the thumbnail existed.

Returns the path of the thumbnail generated. Creates directories if they don't exist.

If the thumbnail cannot be generated and *use_fail_appname* is given, a failure info file will be generated, associated to the given app name so it is not needlessly retried.

Parameters

- **src** (*str*) – path of the source file. Must be an image file. Cannot be a URL.
- **size** – desired size of thumbnail. Can be any of 'large', 256 for large thumbnails or 'normal', 128 for small thumbnails.
- **moreinfo** (*dict*) – optional additional key/values metadata to store in the thumbnail file.
- **use_fail_appname** (*str*) – app name to use when creating a failure info.

Returns the path of the thumbnail, or None if it couldn't be generated

Return type str

vignette.**put_thumbnail**(*src*, *size*, *thumb*, *mtime=None*, *moreinfo=None*)
Put a thumbnail into the store.

This method is typically used for thumbnailing non-image files (like PDFs, videos) or non-local files.

The application creates the thumbnail image on its own, and pushes the thumbnail to the store with this function. The thumbnail is moved to the correct place and the mandatory metadata is set.

Parameters

- **src** (*str*) – the URL or path of the source file being thumbnailed.
- **size** – desired size of thumbnail. Can be any of 'large', 256 for large thumbnails or 'normal', 128 for small thumbnails.
- **thumb** – path of the thumbnail created by the app. This file will be moved to the target. It is advised to use *create_temp* for obtaining a file path.
- **mtime** (*int*) – mtime of the source file. Optional only if *src* is a local file.
- **moreinfo** (*dict*) – additional optional key/values to store in the thumbnail file.

Returns the path where the thumbnail has been moved.

Return type str

vignette.**put_fail**(*src*, *appname*, *mtime=None*, *moreinfo=None*)
Create a failed thumbnail info file.

Creates directories if they don't exist.

When the app tries to generate the thumbnail on its own (to use with *put_thumbnail*) and it fails, the app should use this function to indicate the generation failed and it should not retry (unless the file has been modified).

Parameters

- **src** (*str*) – the URL or path of the source file thumbnailed.
- **mtime** (*int*) – mtime of the source file. Optional only if *src* is a local file.
- **moreinfo** (*dict*) – additional optional key/values to store in the thumbnail file.

Returns path of the failed info file

Return type str

vignette.**is_thumbnail_failed**(*src*, *appname*, *mtime=None*)
Determine whether there exists a fail-file or not.

If a fail-file was generated for file *src* by app *appname*, but the stored mtime in the fail-file is different than the *mtime* argument, the fail-file is considered obsolete so it is ignored and *False* is returned.

This function does not check if a valid thumbnail exists for *src*, it only verifies if a fail-file was created for *src* by *appname*.

See *try_get_thumbnail* for finding if a valid thumbnail exists.

Parameters

- **src** (*str*) – the URL or path of the source file.
- **appname** (*str*) – name of the app
- **mtime** (*int*) – mtime of the source file. Optional only if *src* is a local file.

Return type bool

vignette.**create_temp**(*size*)
Create a temporary file in the thumbnail cache directory.

Return a file path with a random name (with ".png" suffix) in the cache directory for *size*. Should be used by backends to provide UNIX atomic-rename semantics.

Can also be used by apps generating thumbnails on their own (typically for non-image files like PDFs), to then call *put_thumbnail*.

As with function tempfile.mkstemp, the returned file path exists but is guaranteed to be new, so the file can be written to safely.

Parameters **size** – desired size of thumbnail. Can be any of 'large', 256 for large thumbnails or 'normal', 128 for small thumbnails.

Return type str

vignette.**makedirs**()
Create cache directories.

vignette.**KEY_WIDTH = u'Thumb::Image::Width'**
Optional thumbnail metadata key for source image width.

vignette.**KEY_HEIGHT = u'Thumb::Image::Height'**
Optional thumbnail metadata key for source image height.

vignette.**KEY_SIZE = u'Thumb::Size'**
Optional thumbnail metadata key for source file size.

vignette.**KEY_MIME = u'Thumb::Mimetype'**
Optional thumbnail metadata key for source file MIME type.

vignette.**KEY_DOC_PAGES = u'Thumb::Document::Pages'**
Optional thumbnail metadata key for source document number of pages.

vignette.**KEY_MOVIE_LENGTH = u'Thumb::Movie::Length'**
Optional thumbnail metadata key for source video duration (in seconds).

Python Module Index

v

Index