

---

# **osxphotos**

***Release 0.42.9***

**Rhet Turnbull**

**Apr 20, 2021**



# CONTENTS

<b>1</b>	<b>OSXPhotos</b>	<b>1</b>
1.1	What is osxphotos?	1
1.2	Supported operating systems	1
1.3	Installation	1
1.3.1	Installation using pipx	1
1.3.2	Installation using pip	2
1.3.3	Installation from git repository	2
1.4	Command Line Usage	2
1.4.1	Command line examples	3
1.4.1.1	export all photos to ~/Desktop/export group in folders by date created	3
1.4.1.2	find all photos with keyword “Kids” and output results to json file named results.json:	3
1.4.1.3	export photos to file structure based on 4-digit year and full name of month of photo’s creation date:	3
1.4.1.4	export photos to file structure based on 4-digit year of photo’s creation date and add keywords for media type and labels (labels are only available on Photos 5 and higher):	3
1.4.1.5	export default library using ‘country name/year’ as output directory (but use “No-Country/year” if country not specified), add persons, album names, and year as keywords, write exif metadata to files when exporting, update only changed files, print verbose output	3
1.5	Example uses of the package	4
1.6	Package Interface	6
1.6.1	osxphotos command line interface (CLI)	6
1.6.1.1	osxphotos	6
1.6.2	osxphotos package	22
1.6.2.1	osxphotos module	22
<b>2</b>	<b>Indices and tables</b>	<b>35</b>
	<b>Index</b>	<b>37</b>



## OSXPHOTOS

### 1.1 What is osxphotos?

OSXPhotos provides both the ability to interact with and query Apple's Photos.app library on macOS directly from your python code as well as a very flexible command line interface (CLI) app for exporting photos. You can query the Photos library database – for example, file name, file path, and metadata such as keywords/tags, persons/faces, albums, etc. You can also easily export both the original and edited photos.

### 1.2 Supported operating systems

Only works on macOS (aka Mac OS X). Tested on macOS Sierra (10.12.6) until macOS Catalina (10.15.7). Beta support for macOS Big Sur (10.16.01/11.01).

This package will read Photos databases for any supported version on any supported macOS version. E.g. you can read a database created with Photos 5.0 on MacOS 10.15 on a machine running macOS 10.12 and vice versa.

Requires python  $\geq 3.7$ .

### 1.3 Installation

If you are new to python and just want to use the command line application, I recommend you to install using pipx. See other advanced options below.

#### 1.3.1 Installation using pipx

If you aren't familiar with installing python applications, I recommend you install `osxphotos` with `pipx`. If you use `pipx`, you will not need to create a virtual environment as `pipx` takes care of this. The easiest way to do this on a Mac is to use `homebrew`:

- Open Terminal (search for Terminal in Spotlight or look in Applications/Utilities)
- Install homebrew according to instructions at <https://brew.sh/>
- Type the following into Terminal: `brew install pipx`
- Then type this: `pipx install osxphotos`
- Now you should be able to run `osxphotos` by typing: `osxphotos`

### 1.3.2 Installation using pip

You can also install directly from [pypi](#):

```
pip install osxphotos
```

### 1.3.3 Installation from git repository

OSXPhotos uses [setuptools](#), thus simply run:

```
git clone https://github.com/RhetTbull/osxphotos.git
cd osxphotos
python3 setup.py install
```

I recommend you create a [virtual environment](#) before installing osxphotos.

**WARNING** The git repo for this project is very large (> 1GB) because it contains multiple Photos libraries used for testing on different versions of macOS. If you just want to use the osxphotos package in your own code, I recommend you install the latest version from [PyPI](#) which does not include all the test libraries. If you just want to use the command line utility, you can download a pre-built executable of the latest [release](#) or you can install via `pip` which also installs the command line app. If you aren't comfortable with running python on your Mac, start with the pre-built executable or `pipx` as described above.

## 1.4 Command Line Usage

This package will install a command line utility called `osxphotos` that allows you to query the Photos database and export photos. Alternatively, you can also run the command line utility like this: `python3 -m osxphotos`

```
> osxphotos
Usage: osxphotos [OPTIONS] COMMAND [ARGS]...

Options:
  --db <Photos database path> Specify Photos database path. Path to Photos
                             library/database can be specified using either
                             --db or directly as PHOTOS_LIBRARY positional
                             argument. If neither --db or PHOTOS_LIBRARY
                             provided, will attempt to find the library to
                             use in the following order: 1. last opened
                             library, 2. system library, 3.
                             ~/Pictures/Photos Library.photoslibrary
  --json                      Print output in JSON format.
  -v, --version               Show the version and exit.
  -h, --help                  Show this message and exit.

Commands:
  about      Print information about osxphotos including license.
  albums     Print out albums found in the Photos library.
  dump       Print list of all photos & associated info from the Photos...
  export     Export photos from the Photos database.
  help       Print help; for help on commands: help <command>.
  info       Print out descriptive info of the Photos library database.
  keywords   Print out keywords found in the Photos library.
  labels     Print out image classification labels found in the Photos...
  list       Print list of Photos libraries found on the system.
```

(continues on next page)

(continued from previous page)

persons	Print out persons (faces) found <b>in</b> the Photos library.
places	Print out places found <b>in</b> the Photos library.
query	Query the Photos database using <b>1 or more</b> search options; <b>if...</b>

To get help on a specific command, use `osxphotos help <command_name>`

## 1.4.1 Command line examples

### 1.4.1.1 export all photos to ~/Desktop/export group in folders by date created

```
osxphotos export --export-by-date ~/Pictures/Photos\ Library.photoslibrary
~/Desktop/export
```

**Note:** Photos library/database path can also be specified using `--db` option:

```
osxphotos export --export-by-date --db ~/Pictures/Photos\ Library.
photoslibrary ~/Desktop/export
```

### 1.4.1.2 find all photos with keyword “Kids” and output results to json file named results.json:

```
osxphotos query --keyword Kids --json ~/Pictures/Photos\ Library.photoslibrary
>results.json
```

### 1.4.1.3 export photos to file structure based on 4-digit year and full name of month of photo’s creation date:

```
osxphotos export ~/Desktop/export --directory "{created.year}/{created.month}"
(by default, it will attempt to use the system library)
```

### 1.4.1.4 export photos to file structure based on 4-digit year of photo’s creation date and add keywords for media type and labels (labels are only available on Photos 5 and higher):

```
osxphotos export ~/Desktop/export --directory "{created.year}"
--keyword-template "{label}" --keyword-template "{media_type}"
```

### 1.4.1.5 export default library using ‘country name/year’ as output directory (but use “NoCountry/year” if country not specified), add persons, album names, and year as keywords, write exif metadata to files when exporting, update only changed files, print verbose output

```
osxphotos export ~/Desktop/export --directory "{place.name.country,NoCountry}/{created.year}"
--person-keyword --album-keyword --keyword-template "{created.year}" --exiftool --update --verbose
```

## 1.5 Example uses of the package

```

""" Simple usage of the package """
import osxphotos

def main():
    photosdb = osxphotos.PhotosDB()
    print(photosdb.keywords)
    print(photosdb.persons)
    print(photosdb.album_names)

    print(photosdb.keywords_as_dict)
    print(photosdb.persons_as_dict)
    print(photosdb.albums_as_dict)

    # find all photos with Keyword = Foo and containing John Smith
    photos = photosdb.photos(keywords=["Foo"], persons=["John Smith"])

    # find all photos that include Alice Smith but do not contain the keyword Bar
    photos = [p for p in photosdb.photos(persons=["Alice Smith"])
               if p not in photosdb.photos(keywords=["Bar"])]
    for p in photos:
        print(
            p.uuid,
            p.filename,
            p.original_filename,
            p.date,
            p.description,
            p.title,
            p.keywords,
            p.albums,
            p.persons,
            p.path,
        )

if __name__ == "__main__":
    main()

```

```

""" Export all photos to specified directory using album names as folders
    If file has been edited, also export the edited version,
    otherwise, export the original version
    This will result in duplicate photos if photo is in more than album """

import os.path
import pathlib
import sys

import click
from pathvalidate import is_valid_filepath, sanitize_filepath

import osxphotos

@click.command()
@click.argument("export_path", type=click.Path(exists=True))
@click.option(
    "--default-album",

```

(continues on next page)



(continued from previous page)

```

    help="Default folder for photos with no album. Defaults to 'unfiled'",
    default="unfiled",
)
@click.option(
    "--library-path",
    help="Path to Photos library, default to last used library",
    default=None,
)
def export(export_path, default_album, library_path):
    export_path = os.path.expanduser(export_path)
    library_path = os.path.expanduser(library_path) if library_path else None

    if library_path is not None:
        photosdb = osxphotos.PhotosDB(library_path)
    else:
        photosdb = osxphotos.PhotosDB()

    photos = photosdb.photos()

    for p in photos:
        if not p.ismissing:
            albums = p.albums
            if not albums:
                albums = [default_album]
            for album in albums:
                click.echo(f"exporting {p.filename} in album {album}")

                # make sure no invalid characters in destination path (could be in_
↪album name)
                album_name = sanitize_filepath(album, platform="auto")

                # create destination folder, if necessary, based on album name
                dest_dir = os.path.join(export_path, album_name)

                # verify path is a valid path
                if not is_valid_filepath(dest_dir, platform="auto"):
                    sys.exit(f"Invalid filepath {dest_dir}")

                # create destination dir if needed
                if not os.path.isdir(dest_dir):
                    os.makedirs(dest_dir)

                # export the photo
                if p.hasadjustments:
                    # export edited version
                    exported = p.export(dest_dir, edited=True)
                    edited_name = pathlib.Path(p.path_edited).name
                    click.echo(f"Exported {edited_name} to {exported}")
                # export unedited version
                exported = p.export(dest_dir)
                click.echo(f"Exported {p.filename} to {exported}")
            else:
                click.echo(f"Skipping missing photo: {p.filename}")

if __name__ == "__main__":
    export() # pylint: disable=no-value-for-parameter

```

## 1.6 Package Interface

Reference full documentation on [GitHub](#)

### 1.6.1 osxphotos command line interface (CLI)

#### 1.6.1.1 osxphotos

```
osxphotos [OPTIONS] COMMAND [ARGS]...
```

#### Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either **--db** or directly as **PHOTOS\_LIBRARY** positional argument. If neither **--db** or **PHOTOS\_LIBRARY** provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. ~/Pictures/Photos Library.photoslibrary

**--json**

Print output in JSON format.

**-v, --version**

Show the version and exit.

#### about

Print information about osxphotos including license.

```
osxphotos about [OPTIONS]
```

#### albums

Print out albums found in the Photos library.

```
osxphotos albums [OPTIONS] [PHOTOS_LIBRARY]...
```

#### Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either **--db** or directly as **PHOTOS\_LIBRARY** positional argument. If neither **--db** or **PHOTOS\_LIBRARY** provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. ~/Pictures/Photos Library.photoslibrary

**--json**

Print output in JSON format.

## Arguments

### PHOTOS\_LIBRARY

Optional argument(s)

## dump

Print list of all photos & associated info from the Photos library.

```
osxphotos dump [OPTIONS] [PHOTOS_LIBRARY]...
```

## Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either `--db` or directly as `PHOTOS_LIBRARY` positional argument. If neither `--db` or `PHOTOS_LIBRARY` provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. `~/Pictures/Photos Library.photoslibrary`

**--json**

Print output in JSON format.

**--deleted**

Include photos from the 'Recently Deleted' folder.

**--deleted-only**

Include only photos from the 'Recently Deleted' folder.

## Arguments

### PHOTOS\_LIBRARY

Optional argument(s)

## export

Export photos from the Photos database. Export path `DEST` is required. Optionally, query the Photos database using 1 or more search options; if more than one option is provided, they are treated as "AND" (e.g. search for photos matching all options). If no query options are provided, all photos will be exported. By default, all versions of all photos will be exported including edited versions, live photo movies, burst photos, and associated raw images. See `--skip-edited`, `--skip-live`, `--skip-bursts`, and `--skip-raw` options to modify this behavior.

```
osxphotos export [OPTIONS] [PHOTOS_LIBRARY]... DEST
```

## Options

- db** <Photos database path>  
Specify Photos database path. Path to Photos library/database can be specified using either `--db` or directly as `PHOTOS_LIBRARY` positional argument. If neither `--db` or `PHOTOS_LIBRARY` provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. `~/Pictures/Photos Library.photoslibrary`
- V, --verbose**  
Print verbose output.
- keyword** <KEYWORD>  
Search for photos with keyword KEYWORD. If more than one keyword, treated as “OR”, e.g. find photos matching any keyword
- person** <PERSON>  
Search for photos with person PERSON. If more than one person, treated as “OR”, e.g. find photos matching any person
- album** <ALBUM>  
Search for photos in album ALBUM. If more than one album, treated as “OR”, e.g. find photos matching any album
- folder** <FOLDER>  
Search for photos in an album in folder FOLDER. If more than one folder, treated as “OR”, e.g. find photos in any FOLDER. Only searches top level folders (e.g. does not look at subfolders)
- name** <FILENAME>  
Search for photos with filename matching FILENAME. If more than one `--name` options is specified, they are treated as “OR”, e.g. find photos matching any FILENAME.
- uuid** <UUID>  
Search for photos with UUID(s).
- uuid-from-file** <FILE>  
Search for photos with UUID(s) loaded from FILE. Format is a single UUID per line. Lines preceded with `#` are ignored.
- title** <TITLE>  
Search for TITLE in title of photo.
- no-title**  
Search for photos with no title.
- description** <DESC>  
Search for DESC in description of photo.
- no-description**  
Search for photos with no description.
- place** <PLACE>  
Search for PLACE in photo’s reverse geolocation info
- no-place**  
Search for photos with no associated place name info (no reverse geolocation info)
- label** <LABEL>  
Search for photos with image classification label LABEL (Photos 5 only). If more than one label, treated as “OR”, e.g. find photos matching any label
- uti** <UTI>  
Search for photos whose uniform type identifier (UTI) matches UTI

**-i, --ignore-case**  
Case insensitive search for title, description, place, keyword, person, or album.

**--edited**  
Search for photos that have been edited.

**--external-edit**  
Search for photos edited in external editor.

**--favorite**  
Search for photos marked favorite.

**--not-favorite**  
Search for photos not marked favorite.

**--hidden**  
Search for photos marked hidden.

**--not-hidden**  
Search for photos not marked hidden.

**--shared**  
Search for photos in shared iCloud album (Photos 5 only).

**--not-shared**  
Search for photos not in shared iCloud album (Photos 5 only).

**--burst**  
Search for photos that were taken in a burst.

**--not-burst**  
Search for photos that are not part of a burst.

**--live**  
Search for Apple live photos

**--not-live**  
Search for photos that are not Apple live photos.

**--portrait**  
Search for Apple portrait mode photos.

**--not-portrait**  
Search for photos that are not Apple portrait mode photos.

**--screenshot**  
Search for screenshot photos.

**--not-screenshot**  
Search for photos that are not screenshot photos.

**--slow-mo**  
Search for slow motion videos.

**--not-slow-mo**  
Search for photos that are not slow motion videos.

**--time-lapse**  
Search for time lapse videos.

**--not-time-lapse**  
Search for photos that are not time lapse videos.

**--hdr**  
Search for high dynamic range (HDR) photos.

**--not-hdr**  
Search for photos that are not HDR photos.

**--selfie**  
Search for selfies (photos taken with front-facing cameras).

**--not-selfie**  
Search for photos that are not selfies.

**--panorama**  
Search for panorama photos.

**--not-panorama**  
Search for photos that are not panoramas.

**--has-raw**  
Search for photos with both a jpeg and raw version

**--only-movies**  
Search only for movies (default searches both images and movies).

**--only-photos**  
Search only for photos/images (default searches both images and movies).

**--from-date** <from\_date>  
Search by item start date, e.g. 2000-01-12T12:00:00, 2001-01-12T12:00:00-07:00, or 2000-12-31 (ISO 8601 with/without timezone).

**--to-date** <to\_date>  
Search by item end date, e.g. 2000-01-12T12:00:00, 2001-01-12T12:00:00-07:00, or 2000-12-31 (ISO 8601 with/without timezone).

**--from-time** <from\_time>  
Search by item start time of day, e.g. 12:00, or 12:00:00.

**--to-time** <to\_time>  
Search by item end time of day, e.g. 12:00 or 12:00:00.

**--has-comment**  
Search for photos that have comments.

**--no-comment**  
Search for photos with no comments.

**--has-likes**  
Search for photos that have likes.

**--no-likes**  
Search for photos with no likes.

**--is-reference**  
Search for photos that were imported as referenced files (not copied into Photos library).

**--in-album**  
Search for photos that are in one or more albums.

**--not-in-album**  
Search for photos that are not in any albums.

**--min-size** <SIZE>  
Search for photos with size >= SIZE bytes. The size evaluated is the photo's original size (when imported to

Photos). Size may be specified as integer bytes or using SI or NIST units. For example, the following are all valid and equivalent sizes: '1048576' '1.048576MB', '1 MiB'.

**--max-size** <SIZE>

Search for photos with size <= SIZE bytes. The size evaluated is the photo's original size (when imported to Photos). Size may be specified as integer bytes or using SI or NIST units. For example, the following are all valid and equivalent sizes: '1048576' '1.048576MB', '1 MiB'.

**--regex** <REGEX TEMPLATE>

Search for photos where TEMPLATE matches regular expression REGEX. For example, to find photos in an album that begins with 'Beach': '--regex "'^Beach"' "{album}"'. You may specify more than one regular expression match by repeating '--regex' with different arguments.

**--query-eval** <CRITERIA>

Evaluate CRITERIA to filter photos. CRITERIA will be evaluated in context of the following python list comprehension: *photos = [photo for photo in photos if CRITERIA]* where photo represents a PhotoInfo object. For example: *--query-eval photo.favorite* returns all photos that have been favorited and is equivalent to *--favorite*. You may specify more than one CRITERIA by using *--query-eval* multiple times. CRITERIA must be a valid python expression. See <https://rhettbull.github.io/osxphotos/> for additional documentation on the PhotoInfo class.

**--missing**

Export only photos missing from the Photos library; must be used with *--download-missing*.

**--deleted**

Include photos from the 'Recently Deleted' folder.

**--deleted-only**

Include only photos from the 'Recently Deleted' folder.

**--update**

Only export new or updated files. See notes below on export and *--update*.

**--ignore-signature**

When used with '*--update*', ignores file signature when updating files. This is useful if you have processed or edited exported photos changing the file signature (size & modification date). In this case, '*--update*' would normally re-export the processed files but with '*--ignore-signature*', files which exist in the export directory will not be re-exported. If used with '*--sidecar*', '*--ignore-signature*' has the following behavior: 1) if the metadata (in Photos) that went into the sidecar did not change, the sidecar will not be updated; 2) if the metadata (in Photos) that went into the sidecar did change, a new sidecar is written but a new image file is not; 3) if a sidecar does not exist for the photo, a sidecar will be written whether or not the photo file was written or updated.

**--only-new**

If used with *--update*, ignores any previously exported files, even if missing from the export folder and only exports new files that haven't previously been exported.

**--dry-run**

Dry run (test) the export but don't actually export any files; most useful with *--verbose*.

**--export-as-hardlink**

Hardlink files instead of copying them. Cannot be used with *--exiftool* which creates copies of the files with embedded EXIF data. Note: on APFS volumes, files are cloned when exporting giving many of the same advantages as hardlinks without having to use *--export-as-hardlink*.

**--touch-file**

Sets the file's modification time to match photo date.

**--overwrite**

Overwrite existing files. Default behavior is to add (1), (2), etc to filename if file already exists. Use this with caution as it may create name collisions on export. (e.g. if two files happen to have the same name)

**--retry <RETRY>**

Automatically retry export up to RETRY times if an error occurs during export. This may be useful with network drives that experience intermittent errors.

**--export-by-date**

Automatically create output folders to organize photos by date created (e.g. DEST/2019/12/20/photoname.jpg).

**--skip-edited**

Do not export edited version of photo if an edited version exists.

**--skip-original-if-edited**

Do not export original if there is an edited version (exports only the edited version).

**--skip-bursts**

Do not export all associated burst images in the library if a photo is a burst photo.

**--skip-live**

Do not export the associated live video component of a live photo.

**--skip-raw**

Do not export associated raw images of a RAW+JPEG pair. Note: this does not skip raw photos if the raw photo does not have an associated jpeg image (e.g. the raw file was imported to Photos without a jpeg preview).

**--current-name**

Use photo's current filename instead of original filename for export. Note: Starting with Photos 5, all photos are renamed upon import. By default, photos are exported with the the original name they had before import.

**--convert-to-jpeg**

Convert all non-jpeg images (e.g. raw, HEIC, PNG, etc) to JPEG upon export. Only works if your Mac has a GPU.

**--jpeg-quality <jpeg\_quality>**

Value in range 0.0 to 1.0 to use with --convert-to-jpeg. A value of 1.0 specifies best quality, a value of 0.0 specifies maximum compression. Defaults to 1.0

**--download-missing**

Attempt to download missing photos from iCloud. The current implementation uses Applescript to interact with Photos to export the photo which will force Photos to download from iCloud if the photo does not exist on disk. This will be slow and will require internet connection. This obviously only works if the Photos library is synched to iCloud. Note: --download-missing does not currently export all burst images; only the primary photo will be exported—associated burst images will be skipped.

**--sidecar <FORMAT>**

Create sidecar for each photo exported; valid FORMAT values: xmp, json, exiftool; --sidecar xmp: create XMP sidecar used by Digikam, Adobe Lightroom, etc. The sidecar file is named in format photoname.ext.xmp The XMP sidecar exports the following tags: Description, Title, Keywords/Tags, Subject (set to Keywords + Person-InImage), PersonInImage, CreateDate, ModifyDate, GPSLongitude, Face Regions (Metadata Working Group and Microsoft Photo). --sidecar json: create JSON sidecar useable by exiftool (<https://exiftool.org/>) The sidecar file can be used to apply metadata to the file with exiftool, for example: "exiftool -j=photoname.jpg.json photoname.jpg" The sidecar file is named in format photoname.ext.json; format includes tag groups (equivalent to running 'exiftool -G -j'). --sidecar exiftool: create JSON sidecar compatible with output of 'exiftool -j'. Unlike '--sidecar json', '--sidecar exiftool' does not export tag groups. Sidecar filename is in format photoname.ext.json; For a list of tags exported in the JSON and exiftool sidecar, see '--exiftool'. See also '--ignore-signature'.

**Options** xmp | json | exiftool

**--sidecar-drop-ext**

Drop the photo's extension when naming sidecar files. By default, sidecar files are named in format 'photo\_filename.photo\_ext.sidecar\_ext', e.g. 'IMG\_1234.JPG.xmp'. Use '--sidecar-drop-ext' to ignore the



photo extension. Resulting sidecar files will have name in format ‘IMG\_1234.xmp’. Warning: this may result in sidecar filename collisions if there are files of different types but the same name in the output directory, e.g. ‘IMG\_1234.JPG’ and ‘IMG\_1234.MOV’.

#### **--exiftool**

Use exiftool to write metadata directly to exported photos. To use this option, exiftool must be installed and in the path. exiftool may be installed from <https://exiftool.org/>. Cannot be used with `--export-as-hardlink`. Writes the following metadata: EXIF:ImageDescription, XMP:Description (see also `--description-template`); XMP:Title; XMP:TagsList, IPTC:Keywords, XMP:Subject (see also `--keyword-template`, `--person-keyword`, `--album-keyword`); XMP:PersonInImage; EXIF:GPSLatitudeRef; EXIF:GPSLongitudeRef; EXIF:GPSLatitude; EXIF:GPSLongitude; EXIF:GPSPosition; EXIF:DateTimeOriginal; EXIF:OffsetTimeOriginal; EXIF:ModifyDate (see `--ignore-date-modified`); IPTC:DateCreated; IPTC:TimeCreated; (video files only): QuickTime:CreationDate; QuickTime:CreateDate; QuickTime:ModifyDate (see also `--ignore-date-modified`); QuickTime:GPSCoordinates; UserData:GPSCoordinates.

#### **--exiftool-path** <EXIFTOOL\_PATH>

Optionally specify path to exiftool; if not provided, will look for exiftool in \$PATH.

#### **--exiftool-option** <OPTION>

Optional flag/option to pass to exiftool when using `--exiftool`. For example, `--exiftool-option '-m'` to ignore minor warnings. Specify these as you would on the exiftool command line. See exiftool docs at [https://exiftool.org/exiftool\\_pod.html](https://exiftool.org/exiftool_pod.html) for full list of options. More than one option may be specified by repeating the option, e.g. `--exiftool-option '-m' --exiftool-option '-F'`.

#### **--exiftool-merge-keywords**

Merge any keywords found in the original file with keywords used for ‘`--exiftool`’ and ‘`--sidecar`’.

#### **--exiftool-merge-persons**

Merge any persons found in the original file with persons used for ‘`--exiftool`’ and ‘`--sidecar`’.

#### **--ignore-date-modified**

If used with `--exiftool` or `--sidecar`, will ignore the photo modification date and set EXIF:ModifyDate to EXIF:DateTimeOriginal; this is consistent with how Photos handles the EXIF:ModifyDate tag.

#### **--person-keyword**

Use person in image as keyword/tag when exporting metadata.

#### **--album-keyword**

Use album name as keyword/tag when exporting metadata.

#### **--keyword-template** <TEMPLATE>

For use with `--exiftool`, `--sidecar`; specify a template string to use as keyword in the form ‘{name,DEFAULT}’. This is the same format as `--directory`. For example, if you wanted to add the full path to the folder and album photo is contained in as a keyword when exporting you could specify `--keyword-template "{folder_album}"`. You may specify more than one template, for example `--keyword-template "{folder_album}" --keyword-template "{created.year}"`. See ‘`--replace-keywords`’ and Templating System below.

#### **--replace-keywords**

Replace keywords with any values specified with `--keyword-template`. By default, `--keyword-template` will add keywords to any keywords already associated with the photo. If `--replace-keywords` is specified, values from `--keyword-template` will replace any existing keywords instead of adding additional keywords.

#### **--description-template** <TEMPLATE>

For use with `--exiftool`, `--sidecar`; specify a template string to use as description in the form ‘{name,DEFAULT}’. This is the same format as `--directory`. For example, if you wanted to append ‘exported with osxphotos on [today’s date]’ to the description, you could specify `--description-template "{descr} exported with osxphotos on {today.date}"`. See Templating System below.

**--finder-tag-template** <TEMPLATE>

Set MacOS Finder tags to TEMPLATE. These tags can be searched in the Finder or Spotlight with 'tag:tagname' format. For example, '--finder-tag-template "{label}"' to set Finder tags to photo labels. You may specify multiple TEMPLATE values by using '--finder-tag-template' multiple times. See also '--finder-tag-keywords and Extended Attributes below.'

**--finder-tag-keywords**

Set MacOS Finder tags to keywords; any keywords specified via '--keyword-template', '--person-keyword', etc. will also be used as Finder tags. See also '--finder-tag-template and Extended Attributes below.'

**--xattr-template** <ATTRIBUTE TEMPLATE>

Set extended attribute ATTRIBUTE to TEMPLATE value. Valid attributes are: 'authors', 'comment', 'copyright', 'description', 'findercomment', 'headline', 'keywords'. For example, to set Finder comment to the photo's title and description: '--xattr-template findercomment "{title}; {descr}" See Extended Attributes below for additional details on this option.

**--directory** <DIRECTORY>

Optional template for specifying name of output directory in the form '{name,DEFAULT}'. See below for additional details on templating system.

**--filename** <FILENAME>

Optional template for specifying name of output file in the form '{name,DEFAULT}'. File extension will be added automatically—do not include an extension in the FILENAME template. See below for additional details on templating system.

**--jpeg-ext** <EXTENSION>

Specify file extension for JPEG files. Photos uses .jpeg for edited images but many images are imported with .jpg or .JPG which can result in multiple different extensions used for JPEG files upon export. Use --jpeg-ext to specify a single extension to use for all exported JPEG images. Valid values are jpeg, jpg, JPEG, JPG; e.g. '--jpeg-ext jpg' to use '.jpg' for all JPEGs.

**Options** jpeg | jpg | JPEG | JPG

**--strip**

Optionally strip leading and trailing whitespace from any rendered templates. For example, if --filename template is "{title,} {original\_name}" and image has no title, resulting file would have a leading space but if used with --strip, this will be removed.

**--edited-suffix** <SUFFIX>

Optional suffix template for naming edited photos. Default name for edited photos is in form 'photoname\_edited.ext'. For example, with '--edited-suffix \_bearbeiten', the edited photo would be named 'photoname\_bearbeiten.ext'. The default suffix is '\_edited'. Multi-value templates (see Templating System) are not permitted with --edited-suffix.

**--original-suffix** <SUFFIX>

Optional suffix template for naming original photos. Default name for original photos is in form 'filename.ext'. For example, with '--original-suffix \_original', the original photo would be named 'filename\_original.ext'. The default suffix is '' (no suffix). Multi-value templates (see Templating System) are not permitted with --original-suffix.

**--use-photos-export**

Force the use of AppleScript or PhotoKit to export even if not missing (see also '--download-missing' and '--use-photokit').

**--use-photokit**

Use with '--download-missing' or '--use-photos-export' to use direct Photos interface instead of AppleScript to export. Highly experimental alpha feature; does not work with iTerm2 (use with Terminal.app). This is faster and more reliable than the default AppleScript interface.

**--report** <path to export report>

Write a CSV formatted report of all files that were exported.

**--cleanup**

Cleanup export directory by deleting any files which were not included in this export set. For example, photos which had previously been exported and were subsequently deleted in Photos. **WARNING:** `--cleanup` will delete *any* files in the export directory that were not exported by osxphotos, for example, your own scripts or other files. Be sure this is what you intend before using `--cleanup`. Use `--dry-run` with `--cleanup` first if you're not certain.

**--exportdb** <EXPORTDB\_FILE>

Specify alternate name for database file which stores state information for export and `--update`. If `--exportdb` is not specified, export database will be saved to `osxphotos_export.db` in the export directory. Must be specified as filename only, not a path, as export database will be saved in export directory.

**--load-config** <config file path>

Load options from file as written with `--save-config`. This allows you to save a complex export command to file for later reuse. For example: `osxphotos export <lots of options here> --save-config osxphotos.toml` then `osxphotos export /path/to/export --load-config osxphotos.toml`. If any other command line options are used in conjunction with `--load-config`, they will override the corresponding values in the config file.

**--save-config** <config file path>

Save options to file for use with `--load-config`. File format is TOML.

## Arguments

**PHOTOS\_LIBRARY**

Optional argument(s)

**DEST**

Required argument

## help

Print help; for help on commands: `help <command>`.

```
osxphotos help [OPTIONS] [TOPIC]
```

## Arguments

**TOPIC**

Optional argument

## info

Print out descriptive info of the Photos library database.

```
osxphotos info [OPTIONS] [PHOTOS_LIBRARY]...
```

## Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either `--db` or directly as `PHOTOS_LIBRARY` positional argument. If neither `--db` or `PHOTOS_LIBRARY` provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. `~/Pictures/Photos Library.photoslibrary`

**--json**

Print output in JSON format.

## Arguments

**PHOTOS\_LIBRARY**

Optional argument(s)

## keywords

Print out keywords found in the Photos library.

```
osxphotos keywords [OPTIONS] [PHOTOS_LIBRARY]...
```

## Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either `--db` or directly as `PHOTOS_LIBRARY` positional argument. If neither `--db` or `PHOTOS_LIBRARY` provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. `~/Pictures/Photos Library.photoslibrary`

**--json**

Print output in JSON format.

## Arguments

**PHOTOS\_LIBRARY**

Optional argument(s)

## labels

Print out image classification labels found in the Photos library.

```
osxphotos labels [OPTIONS] [PHOTOS_LIBRARY]...
```

## Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either `--db` or directly as `PHOTOS_LIBRARY` positional argument. If neither `--db` or `PHOTOS_LIBRARY` provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. `~/Pictures/Photos Library.photoslibrary`

**--json**

Print output in JSON format.

## Arguments

**PHOTOS\_LIBRARY**

Optional argument(s)

## list

Print list of Photos libraries found on the system.

```
osxphotos list [OPTIONS]
```

## Options

**--json**

Print output in JSON format.

## persons

Print out persons (faces) found in the Photos library.

```
osxphotos persons [OPTIONS] [PHOTOS_LIBRARY]...
```

## Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either `--db` or directly as `PHOTOS_LIBRARY` positional argument. If neither `--db` or `PHOTOS_LIBRARY` provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. `~/Pictures/Photos Library.photoslibrary`

**--json**

Print output in JSON format.

## Arguments

### PHOTOS\_LIBRARY

Optional argument(s)

## places

Print out places found in the Photos library.

```
osxphotos places [OPTIONS] [PHOTOS_LIBRARY]...
```

## Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either **--db** or directly as **PHOTOS\_LIBRARY** positional argument. If neither **--db** or **PHOTOS\_LIBRARY** provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. ~/Pictures/Photos Library.photoslibrary

**--json**

Print output in JSON format.

## Arguments

### PHOTOS\_LIBRARY

Optional argument(s)

## query

Query the Photos database using 1 or more search options; if more than one option is provided, they are treated as “AND” (e.g. search for photos matching all options).

```
osxphotos query [OPTIONS] [PHOTOS_LIBRARY]...
```

## Options

**--db** <Photos database path>

Specify Photos database path. Path to Photos library/database can be specified using either **--db** or directly as **PHOTOS\_LIBRARY** positional argument. If neither **--db** or **PHOTOS\_LIBRARY** provided, will attempt to find the library to use in the following order: 1. last opened library, 2. system library, 3. ~/Pictures/Photos Library.photoslibrary

**--json**

Print output in JSON format.

**--keyword** <KEYWORD>

Search for photos with keyword **KEYWORD**. If more than one keyword, treated as “OR”, e.g. find photos matching any keyword

**--person** <PERSON>

Search for photos with person **PERSON**. If more than one person, treated as “OR”, e.g. find photos matching any person

---

**--album** <ALBUM>  
Search for photos in album ALBUM. If more than one album, treated as “OR”, e.g. find photos matching any album

**--folder** <FOLDER>  
Search for photos in an album in folder FOLDER. If more than one folder, treated as “OR”, e.g. find photos in any FOLDER. Only searches top level folders (e.g. does not look at subfolders)

**--name** <FILENAME>  
Search for photos with filename matching FILENAME. If more than one --name options is specified, they are treated as “OR”, e.g. find photos matching any FILENAME.

**--uuid** <UUID>  
Search for photos with UUID(s).

**--uuid-from-file** <FILE>  
Search for photos with UUID(s) loaded from FILE. Format is a single UUID per line. Lines preceded with # are ignored.

**--title** <TITLE>  
Search for TITLE in title of photo.

**--no-title**  
Search for photos with no title.

**--description** <DESC>  
Search for DESC in description of photo.

**--no-description**  
Search for photos with no description.

**--place** <PLACE>  
Search for PLACE in photo’s reverse geolocation info

**--no-place**  
Search for photos with no associated place name info (no reverse geolocation info)

**--label** <LABEL>  
Search for photos with image classification label LABEL (Photos 5 only). If more than one label, treated as “OR”, e.g. find photos matching any label

**--uti** <UTI>  
Search for photos whose uniform type identifier (UTI) matches UTI

**-i, --ignore-case**  
Case insensitive search for title, description, place, keyword, person, or album.

**--edited**  
Search for photos that have been edited.

**--external-edit**  
Search for photos edited in external editor.

**--favorite**  
Search for photos marked favorite.

**--not-favorite**  
Search for photos not marked favorite.

**--hidden**  
Search for photos marked hidden.

**--not-hidden**  
Search for photos not marked hidden.

**--shared**  
Search for photos in shared iCloud album (Photos 5 only).

**--not-shared**  
Search for photos not in shared iCloud album (Photos 5 only).

**--burst**  
Search for photos that were taken in a burst.

**--not-burst**  
Search for photos that are not part of a burst.

**--live**  
Search for Apple live photos

**--not-live**  
Search for photos that are not Apple live photos.

**--portrait**  
Search for Apple portrait mode photos.

**--not-portrait**  
Search for photos that are not Apple portrait mode photos.

**--screenshot**  
Search for screenshot photos.

**--not-screenshot**  
Search for photos that are not screenshot photos.

**--slow-mo**  
Search for slow motion videos.

**--not-slow-mo**  
Search for photos that are not slow motion videos.

**--time-lapse**  
Search for time lapse videos.

**--not-time-lapse**  
Search for photos that are not time lapse videos.

**--hdr**  
Search for high dynamic range (HDR) photos.

**--not-hdr**  
Search for photos that are not HDR photos.

**--selfie**  
Search for selfies (photos taken with front-facing cameras).

**--not-selfie**  
Search for photos that are not selfies.

**--panorama**  
Search for panorama photos.

**--not-panorama**  
Search for photos that are not panoramas.



---

**--has-raw**  
Search for photos with both a jpeg and raw version

**--only-movies**  
Search only for movies (default searches both images and movies).

**--only-photos**  
Search only for photos/images (default searches both images and movies).

**--from-date** <from\_date>  
Search by item start date, e.g. 2000-01-12T12:00:00, 2001-01-12T12:00:00-07:00, or 2000-12-31 (ISO 8601 with/without timezone).

**--to-date** <to\_date>  
Search by item end date, e.g. 2000-01-12T12:00:00, 2001-01-12T12:00:00-07:00, or 2000-12-31 (ISO 8601 with/without timezone).

**--from-time** <from\_time>  
Search by item start time of day, e.g. 12:00, or 12:00:00.

**--to-time** <to\_time>  
Search by item end time of day, e.g. 12:00 or 12:00:00.

**--has-comment**  
Search for photos that have comments.

**--no-comment**  
Search for photos with no comments.

**--has-likes**  
Search for photos that have likes.

**--no-likes**  
Search for photos with no likes.

**--is-reference**  
Search for photos that were imported as referenced files (not copied into Photos library).

**--in-album**  
Search for photos that are in one or more albums.

**--not-in-album**  
Search for photos that are not in any albums.

**--min-size** <SIZE>  
Search for photos with size  $\geq$  SIZE bytes. The size evaluated is the photo's original size (when imported to Photos). Size may be specified as integer bytes or using SI or NIST units. For example, the following are all valid and equivalent sizes: '1048576' '1.048576MB', '1 MiB'.

**--max-size** <SIZE>  
Search for photos with size  $\leq$  SIZE bytes. The size evaluated is the photo's original size (when imported to Photos). Size may be specified as integer bytes or using SI or NIST units. For example, the following are all valid and equivalent sizes: '1048576' '1.048576MB', '1 MiB'.

**--regex** <REGEX TEMPLATE>  
Search for photos where TEMPLATE matches regular expression REGEX. For example, to find photos in an album that begins with 'Beach': '-regex "'^Beach"' "{album}"'. You may specify more than one regular expression match by repeating '-regex' with different arguments.

**--query-eval** <CRITERIA>  
Evaluate CRITERIA to filter photos. CRITERIA will be evaluated in context of the following python list comprehension: *photos* = [*photo* for *photo* in *photos* if *CRITERIA*] where *photo* represents a PhotoInfo object.

For example: `-query-eval photo.favorite` returns all photos that have been favorited and is equivalent to `-favorite`. You may specify more than one CRITERIA by using `-query-eval` multiple times. CRITERIA must be a valid python expression. See <https://rhettbull.github.io/osxphotos/> for additional documentation on the PhotoInfo class.

**--deleted**

Include photos from the 'Recently Deleted' folder.

**--deleted-only**

Include only photos from the 'Recently Deleted' folder.

**--missing**

Search for photos missing from disk.

**--not-missing**

Search for photos present on disk (e.g. not missing).

**--cloudasset**

Search for photos that are part of an iCloud library

**--not-cloudasset**

Search for photos that are not part of an iCloud library

**--incloud**

Search for photos that are in iCloud (have been synched)

**--not-incloud**

Search for photos that are not in iCloud (have not been synched)

## Arguments

**PHOTOS\_LIBRARY**

Optional argument(s)

## 1.6.2 osxphotos package

### 1.6.2.1 osxphotos module

**class** `osxphotos.PhotosDB` (*dbfile=None, verbose=None, exiftool=None*)

Processes a Photos.app library database to extract information about photos

**property** `album_info`

return list of AlbumInfo objects for each album in the photos database

**property** `album_info_shared`

return list of AlbumInfo objects for each shared album in the photos database only valid for Photos 5; on Photos <= 4, prints warning and returns empty list

**property** `albums`

return list of albums found in photos database

**property** `albums_as_dict`

return albums as dict of albums, count in reverse sorted order (descending)

**property** `albums_shared`

return list of shared albums found in photos database only valid for Photos 5; on Photos <= 4, prints warning and returns empty list

**property albums\_shared\_as\_dict**

returns shared albums as dict of albums, count in reverse sorted order (descending) valid only on Photos 5; on Photos <= 4, prints warning and returns empty dict

**property db\_path**

returns path to the Photos library database PhotosDB was initialized with

**property db\_version**

return the database version as stored in LiGlobals table

**property folder\_info**

return list FolderInfo objects representing top-level folders in the photos database

**property folders**

return list of top-level folder names in the photos database

**get\_db\_connection()**

Get connection to the working copy of the Photos database

**Returns** tuple of (connection, cursor) to sqlite3 database

**get\_photo(uuid)**

Returns a single photo matching uuid

**Parameters** **uuid** – the UUID of photo to get

**Returns** PhotoInfo instance for photo with UUID matching uuid or None if no match

**property import\_info**

return list of ImportInfo objects for each import session in the database

**property keywords**

return list of keywords found in photos database

**property keywords\_as\_dict**

return keywords as dict of keyword, count in reverse sorted order (descending)

**property labels**

return list of all search info labels found in the library

**property labels\_as\_dict**

count in reverse sorted order (descending)

**Type** return labels as dict of label

**property labels\_normalized**

return list of all normalized search info labels found in the library

**property labels\_normalized\_as\_dict**

count in reverse sorted order (descending)

**Type** return normalized labels as dict of label

**property library\_path**

returns path to the Photos library PhotosDB was initialized with

**property person\_info**

return list of PersonInfo objects for each person in the photos database

**property persons**

return list of persons found in photos database

**property persons\_as\_dict**

return persons as dict of person, count in reverse sorted order (descending)

**photos** (*keywords=None, uuid=None, persons=None, albums=None, images=True, movies=True, from\_date=None, to\_date=None, intrash=False*)

Return a list of PhotoInfo objects If called with no args, returns the entire database of photos If called with args, returns photos matching the args (e.g. keywords, persons, etc.) If more than one arg, returns photos matching all the criteria (e.g. keywords AND persons) If more than one keyword, uuid, persons, albums is passed, they are treated as “OR” criteria e.g. keywords=[“wedding”, “vacation”] returns photos matching either keyword from\_date and to\_date may be either naive or timezone-aware datetime.datetime objects. If naive, timezone will be assumed to be local timezone.

#### Parameters

- **keywords** – list of keywords to search for
- **uuid** – list of UUIDs to search for
- **persons** – list of persons to search for
- **albums** – list of album names to search for
- **images** – if True, returns image files, if False, does not return images; default is True
- **movies** – if True, returns movie files, if False, does not return movies; default is True
- **from\_date** – return photos with creation date  $\geq$  from\_date (datetime.datetime object, default None)
- **to\_date** – return photos with creation date  $\leq$  to\_date (datetime.datetime object, default None)
- **intrash** – if True, returns only images in “Recently deleted items” folder, if False returns only photos that aren’t deleted; default is False

**Returns** list of PhotoInfo objects

**photos\_by\_uuid** (*uuids*)

**Returns a list of photos with UUID in uuids.** Does not generate error if invalid or missing UUID passed. This is faster than using PhotosDB.photos if you have list of UUIDs. Returns photos regardless of intrash state.

**Parameters** **uuid** – list of UUIDs of photos to get

**Returns** list of PhotoInfo instance for photo with UUID matching uuid or [] if no match

**query** (*options: osxphotos.queryoptions.QueryOptions*)  $\rightarrow$  List[osxphotos.photoinfo.photoinfo.PhotoInfo]

Run a query against PhotosDB to extract the photos based on user supplied options

**Parameters** **options** – a QueryOptions instance

**class** osxphotos.PhotoInfo (*db=None, uuid=None, info=None*)

Info about a specific photo, contains all the details about the photo including keywords, persons, albums, uuid, path, etc.

**class** ExifInfo (*flash\_fired: bool, iso: int, metering\_mode: int, sample\_rate: int, track\_format: int, white\_balance: int, aperture: float, bit\_rate: float, duration: float, exposure\_bias: float, focal\_length: float, fps: float, latitude: float, longitude: float, shutter\_speed: float, camera\_make: str, camera\_model: str, codec: str, lens\_model: str*)

EXIF info associated with a photo from the Photos library

**aperture:** float

**bit\_rate:** float

**camera\_make:** str

```

camera_model: str
codec: str
duration: float
exposure_bias: float
flash_fired: bool
focal_length: float
fps: float
iso: int
latitude: float
lens_model: str
longitude: float
metering_mode: int
sample_rate: int
shutter_speed: float
track_format: int
white_balance: int

class ExportResults (exported=None, new=None, updated=None, skipped=None,
                    exif_updated=None, touched=None, converted_to_jpeg=None,
                    sidecar_json_written=None, sidecar_json_skipped=None, side-
                    car_exiftool_written=None, sidecar_exiftool_skipped=None, side-
                    car_xmp_written=None, sidecar_xmp_skipped=None, miss-
                    ing=None, error=None, exiftool_warning=None, exiftool_error=None,
                    xattr_written=None, xattr_skipped=None, deleted_files=None,
                    deleted_directories=None)
    holds export results for export2

    all_files()
        return all filenames contained in results

class ScoreInfo (overall: float, curation: float, promotion: float, highlight_visibility: float,
                behavioral: float, failure: float, harmonious_color: float, immersiveness:
                float, interaction: float, interesting_subject: float, intrusive_object_presence:
                float, lively_color: float, low_light: float, noise: float, pleasant_camera_tilt:
                float, pleasant_composition: float, pleasant_lighting: float, pleasant_pattern:
                float, pleasant_perspective: float, pleasant_post_processing: float, pleas-
                ant_reflection: float, pleasant_symmetry: float, sharply_focused_subject: float,
                tastefully_blurred: float, well_chosen_subject: float, well_framed_subject: float,
                well_timed_shot: float)
    Computed photo score info associated with a photo from the Photos library

    behavioral: float
    curation: float
    failure: float
    harmonious_color: float
    highlight_visibility: float

```

```
immersiveness: float
interaction: float
interesting_subject: float
intrusive_object_presence: float
lively_color: float
low_light: float
noise: float
overall: float
pleasant_camera_tilt: float
pleasant_composition: float
pleasant_lighting: float
pleasant_pattern: float
pleasant_perspective: float
pleasant_post_processing: float
pleasant_reflection: float
pleasant_symmetry: float
promotion: float
sharply_focused_subject: float
tastefully_blurred: float
well_chosen_subject: float
well_framed_subject: float
well_timed_shot: float
```

```
class SearchInfo (photo, normalized=False)
```

Info about search terms such as machine learning labels that Photos knows about a photo

```
property activities
```

returns list of activity names

```
property all
```

return all search info properties in a single list

```
asdict ()
```

return dict of search info

```
property bodies_of_water
```

returns list of body of water names

```
property city
```

returns city/town

```
property country
```

returns country name

```
property holidays
```

returns list of holiday names

---

**property labels**  
return list of labels associated with Photo

**property locality\_names**  
returns list of other locality names

**property media\_types**  
returns list of media types (photo, video, panorama, etc)

**property month**  
returns month name

**property neighborhoods**  
returns list of neighborhoods

**property place\_names**  
returns list of place names

**property season**  
returns season name

**property state**  
returns state name

**property state\_abbreviation**  
returns state abbreviation

**property streets**  
returns list of street names

**property venue\_types**  
returns list of venue types

**property venues**  
returns list of venue names

**property year**  
returns year

**property adjustments**  
Returns AdjustmentsInfo class for adjustment data or None if no adjustments; Photos 5+ only

**property album\_info**  
list of AlbumInfo objects representing albums the photo is contained in

**property albums**  
list of albums picture is contained in

**asdict ()**  
return dict representation

**property burst**  
Returns True if photo is part of a Burst photo set, otherwise False

**property burst\_album\_info**  
If photo is a non-selected burst photo, returns list of AlbumInfo objects representing albums any other photos in the same burst set are contained in, otherwise returns self.album\_info

**property burst\_albums**  
If photo is non-selected burst photo, list of albums any other images in the same burst set are contained in, otherwise returns self.albums

**property burst\_photos**

If photo is a burst photo, returns list of PhotoInfo objects that are part of the same burst photo set; otherwise returns empty list. self is not included in the returned list

**property burst\_selected**

Returns True if photo is a burst photo and has been selected from the burst set by the user, otherwise False

**property comments**

Returns list of Comment objects for any comments on the photo (sorted by date)

**property date**

image creation date as timezone aware datetime object

**property date\_modified**

image modification date as timezone aware datetime object or None if no modification date set

**property date\_trashed**

Date asset was placed in the trash or None

**property description**

long / extended description of picture

**property exif\_info**

Returns an ExifInfo object with the EXIF data for photo Note: the returned EXIF data is the data Photos stores in the database on import; ExifInfo does not provide access to the EXIF info in the actual image file Some or all of the fields may be None Only valid for Photos 5; on earlier database returns None

**property exiftool**

Returns an ExifTool object for the photo requires that exiftool (<https://exiftool.org/>) be installed If exiftool not installed, logs warning and returns None If photo path is missing, returns None

**export** (*dest, \*filename, edited=False, live\_photo=False, raw\_photo=False, export\_as\_hardlink=False, overwrite=False, increment=True, sidecar\_json=False, sidecar\_exiftool=False, sidecar\_xmp=False, use\_photos\_export=False, timeout=120, exiftool=False, use\_albums\_as\_keywords=False, use\_persons\_as\_keywords=False, keyword\_template=None, description\_template=None*)

export photo dest: must be valid destination path (or exception raised) filename: (optional): name of exported picture; if not provided, will use current filename

**NOTE:** if provided, user must ensure file extension (suffix) is correct. For example, if photo is .CR2 file, edited image may be .jpeg. If you provide an extension different than what the actual file is, export will print a warning but will export the photo using the incorrect file extension (unless use\_photos\_export is true, in which case export will use the extension provided by Photos upon export; in this case, an incorrect extension is silently ignored). e.g. to get the extension of the edited photo, reference PhotoInfo.path\_edited

**edited: (boolean, default=False); if True will export the edited version of the photo** (or raise exception if no edited version)

live\_photo: (boolean, default=False); if True, will also export the associated .mov for live photos raw\_photo: (boolean, default=False); if True, will also export the associated RAW photo export\_as\_hardlink: (boolean, default=False); if True, will hardlink files instead of copying them overwrite: (boolean, default=False); if True will overwrite files if they already exist increment: (boolean, default=True); if True, will increment file name until a non-existent name is found

if overwrite=False and increment=False, export will fail if destination file already exists

**sidecar\_json: if set will write a json sidecar with data in format readable by exiftool** sidecar file-name will be dest/filename.json; includes exiftool tag group names (e.g. *exiftool -G -j*)



**sidecar\_exiftool:** if set will write a json sidecar with data in format readable by exiftool sidecar filename will be dest/filename.json; does not include exiftool tag group names (e.g. *exiftool -j*)

**sidecar\_xmp:** if set will write an XMP sidecar with IPTC data sidecar filename will be dest/filename.xmp

**use\_photos\_export:** (boolean, default=False); if True will attempt to export photo via applescript interaction with Photos **timeout:** (int, default=120) timeout in seconds used with use\_photos\_export **exiftool:** (boolean, default = False); if True, will use exiftool to write metadata to export file returns list of full paths to the exported files **use\_albums\_as\_keywords:** (boolean, default = False); if True, will include album names in keywords when exporting metadata with exiftool or sidecar **use\_persons\_as\_keywords:** (boolean, default = False); if True, will include person names in keywords when exporting metadata with exiftool or sidecar **keyword\_template:** (list of strings); list of template strings that will be rendered as used as keywords **description\_template:** string; optional template string that will be rendered for use as photo description

Returns: list of photos exported

```
export2 (dest, *filename, edited=False, live_photo=False, raw_photo=False, export_as_hardlink=False, overwrite=False, increment=True, sidecar=0, sidecar_drop_ext=False, use_photos_export=False, timeout=120, exiftool=False, use_albums_as_keywords=False, use_persons_as_keywords=False, keyword_template=None, description_template=None, update=False, ignore_signature=False, export_db=None, fileutil=<class 'osxphotos.fileutil.FileUtil'>, dry_run=False, touch_file=False, convert_to_jpeg=False, jpeg_quality=1.0, ignore_date_modified=False, use_photokit=False, verbose=None, exiftool_flags=None, merge_exif_keywords=False, merge_exif_persons=False, jpeg_ext=None, persons=True, location=True, replace_keywords=False)
```

export photo, like export but with update and dry\_run options dest: must be valid destination path or exception raised filename: (optional): name of exported picture; if not provided, will use current filename

**NOTE:** if provided, user must ensure file extension (suffix) is correct. For example, if photo is .CR2 file, edited image may be .jpeg. If you provide an extension different than what the actual file is, will export the photo using the incorrect file extension (unless use\_photos\_export is true, in which case export will use the extension provided by Photos upon export. e.g. to get the extension of the edited photo, reference PhotoInfo.path\_edited

**edited:** (boolean, default=False); if True will export the edited version of the photo (or raise exception if no edited version)

**live\_photo:** (boolean, default=False); if True, will also export the associated .mov for live photos **raw\_photo:** (boolean, default=False); if True, will also export the associated RAW photo **export\_as\_hardlink:** (boolean, default=False); if True, will hardlink files instead of copying them **overwrite:** (boolean, default=False); if True will overwrite files if they already exist **increment:** (boolean, default=True); if True, will increment file name until a non-existent name is found

if overwrite=False and increment=False, export will fail if destination file already exists

**sidecar:** bit field: set to one or more of **SIDECAR\_XMP**, **SIDECAR\_JSON**, **SIDECAR\_EXIFTOOL**

**SIDECAR\_JSON:** if set will write a json sidecar with data in format readable by exiftool

sidecar filename will be dest/filename.json; includes exiftool tag group names (e.g. *exiftool -G -j*)

**SIDECAR\_EXIFTOOL:** if set will write a json sidecar with data in format readable by exiftool

sidecar filename will be dest/filename.json; does not include exiftool tag group names (e.g. *exiftool -j*)

**SIDECAR\_XMP: if set will write an XMP sidecar with IPTC data** sidecar filename will be dest/filename.xmp

sidecar\_drop\_ext: (boolean, default=False); if True, drops the photo's extension from sidecar filename (e.g. 'IMG\_1234.json' instead of 'IMG\_1234.JPG.json') use\_photos\_export: (boolean, default=False); if True will attempt to export photo via applescript interaction with Photos timeout: (int, default=120) timeout in seconds used with use\_photos\_export exiftool: (boolean, default = False); if True, will use exiftool to write metadata to export file use\_albums\_as\_keywords: (boolean, default = False); if True, will include album names in keywords when exporting metadata with exiftool or sidecar use\_persons\_as\_keywords: (boolean, default = False); if True, will include person names in keywords when exporting metadata with exiftool or sidecar keyword\_template: (list of strings); list of template strings that will be rendered as used as keywords description\_template: string; optional template string that will be rendered for use as photo description update: (boolean, default=False); if True export will run in update mode, that is, it will

not export the photo if the current version already exists in the destination

ignore\_signature: (bool, default=False), ignore file signature when used with update (look only at filename) export\_db: (ExportDB\_ABC); instance of a class that conforms to ExportDB\_ABC with methods

for getting/setting data related to exported files to compare update state

fileutil: (FileUtilABC); class that conforms to FileUtilABC with various file utilities dry\_run: (boolean, default=False); set to True to run in "dry run" mode touch\_file: (boolean, default=False); if True, sets file's modification time upon photo date convert\_to\_jpeg: boolean; if True, converts non-jpeg images to jpeg jpeg\_quality: float in range 0.0 <= jpeg\_quality <= 1.0. A value of 1.0 specifies use best quality, a value of 0.0 specifies use maximum compression. ignore\_date\_modified: for use with sidecar and exiftool; if True, sets EXIF:ModifyDate to EXIF:DateTimeOriginal even if date\_modified is set verbose: optional callable function to use for printing verbose text during processing; if None (default), does not print output. exiftool\_flags: optional list of flags to pass to exiftool when using exiftool option, e.g ["-m", "-F"] merge\_exif\_keywords: boolean; if True, merged keywords found in file's exif data (requires exiftool) merge\_exif\_persons: boolean; if True, merged persons found in file's exif data (requires exiftool) jpeg\_ext: if set, will use this value for extension on jpegs converted to jpeg with convert\_to\_jpeg; if not set, uses jpeg; do not include the leading "." persons: if True, include persons in exported metadata location: if True, include location in exported metadata replace\_keywords: if True, keyword\_template replaces any keywords, otherwise it's additive

**Returns: ExportResults class** ExportResults has attributes: "exported", "new", "updated", "skipped", "exif\_updated", "touched", "converted\_to\_jpeg", "sidecar\_json\_written", "sidecar\_json\_skipped", "sidecar\_exiftool\_written", "sidecar\_exiftool\_skipped", "sidecar\_xmp\_written", "sidecar\_xmp\_skipped", "missing", "error", "error\_str", "exiftool\_warning", "exiftool\_error",

**Note: to use dry run mode, you must set dry\_run=True and also pass in memory version of export\_db,** and no-op fileutil (e.g. ExportDBInMemory and FileUtilNoOp)

**property external\_edit**

Returns True if picture was edited outside of Photos using external editor

**property face\_info**

list of FaceInfo objects for faces in picture

**property favorite**

True if picture is marked as favorite

**property filename**

filename of the picture

**property has\_raw**

returns True if photo has an associated raw image (that is, it's a RAW+JPEG pair), otherwise False

**property hasadjustments**

True if picture has adjustments / edits

**property hdr**

Returns True if photo is an HDR photo, otherwise False

**property height**

returns height of the current photo version in pixels

**property hidden**

True if picture is hidden

**property import\_info**

ImportInfo object representing import session for the photo or None if no import session

**property incloud**

Returns True if photo is cloud asset and is synched to cloud False if photo is cloud asset and not yet synched to cloud None if photo is not cloud asset

**property intrash**

True if picture is in trash ('Recently Deleted' folder)

**property iscloudasset**

Returns True if photo is a cloud asset (in an iCloud library), otherwise False

**property ismissing**

returns true if photo is missing from disk (which means it's not been downloaded from iCloud) NOTE: the photos.db database uses an asynchronous write-ahead log so changes in Photos

do not immediately get written to disk. In particular, I've noticed that downloading an image from the cloud does not force the database to be updated until something else e.g. an edit, keyword, etc. occurs forcing a database synch The exact process / timing is a mystery to be but be aware that if some photos were recently downloaded from cloud to local storate their status in the database might still show isMissing = 1

**property ismovie**

Returns True if file is a movie, otherwise False

**property isphoto**

Returns True if file is an image, otherwise False

**property israw**

returns True if photo is a raw image. For images with an associated RAW+JPEG pair, see has\_raw

**property isreference**

Returns True if photo is a reference (not copied to the Photos library), otherwise False

**json ()**

Return JSON representation

**property keywords**

list of keywords for picture

**property labels**

returns list of labels applied to photo by Photos image categorization only valid on Photos 5, on older libraries returns empty list

**property labels\_normalized**

returns normalized list of labels applied to photo by Photos image categorization only valid on Photos 5, on older libraries returns empty list

**property likes**

Returns list of Like objects for any likes on the photo (sorted by date)

**property live\_photo**

Returns True if photo is a live photo, otherwise False

**property location**

returns (latitude, longitude) as float in degrees or None

**property orientation**

returns EXIF orientation of the current photo version as int or 0 if current orientation cannot be determined

**property original\_filename**

original filename of the picture Photos 5 mangles filenames upon import

**property original\_filesize**

returns filesize of original photo in bytes as int

**property original\_height**

returns height of the original photo version in pixels

**property original\_orientation**

returns EXIF orientation of the original photo version as int

**property original\_width**

returns width of the original photo version in pixels

**property panorama**

Returns True if photo is a panorama, otherwise False

**property path**

absolute path on disk of the original picture

**property path\_edited**

absolute path on disk of the edited picture

**property path\_live\_photo**

Returns path to the associated video file for a live photo If photo is not a live photo, returns None If photo is missing, returns None

**property path\_raw**

absolute path of associated RAW image or None if there is not one

**property person\_info**

list of PersonInfo objects for person in picture

**property persons**

list of persons in picture

**property place**

Returns PlaceInfo object containing reverse geolocation info

**property portrait**

Returns True if photo is a portrait, otherwise False

**property raw\_original**

returns True if associated raw image and the raw image is selected in Photos via “Use RAW as Original ” otherwise returns False

**render\_template** (*template\_str*, *none\_str*='\_', *path\_sep*=None, *expand\_inplace*=False, *inplace\_sep*=None, *filename*=False, *dirname*=False, *strip*=False)

Renders a template string for PhotoInfo instance using PhotoTemplate

**Parameters**

- **template\_str** – a template string with fields to render
- **none\_str** – a str to use if template field renders to None, default is “\_”.

- **path\_sep** – a single character str to use as path separator when joining fields like folder\_album; if not provided, defaults to os.path.sep
- **expand\_inplace** – expand multi-valued substitutions in-place as a single string instead of returning individual strings
- **inplace\_sep** – optional string to use as separator between multi-valued keywords with expand\_inplace; default is ‘,’
- **filename** – if True, template output will be sanitized to produce valid file name
- **dirname** – if True, template output will be sanitized to produce valid directory name
- **strip** – if True, strips leading/trailing white space from resulting template

**Returns** tuple of list of rendered strings and list of unmatched template values

**Return type** ([rendered\_strings], [unmatched])

**property score**

Computed score information for a photo

**Returns** ScoreInfo instance

**property screenshot**

Returns True if photo is an HDR photo, otherwise False

**property search\_info**

returns SearchInfo object for photo only valid on Photos 5, on older libraries, returns None

**property search\_info\_normalized**

returns SearchInfo object for photo that produces normalized results only valid on Photos 5, on older libraries, returns None

**property selfie**

Returns True if photo is a selfie (front facing camera), otherwise False

**property shared**

returns True if photos is in a shared iCloud album otherwise false Only valid on Photos 5; returns None on older versions

**property slow\_mo**

Returns True if photo is a slow motion video, otherwise False

**property time\_lapse**

Returns True if photo is a time lapse video, otherwise False

**property title**

name / title of picture

**property tzoffset**

timezone offset from UTC in seconds

**property uti**

Returns Uniform Type Identifier (UTI) for the image for example: public.jpeg or com.apple.quicktime-movie

**property uti\_edited**

Returns Uniform Type Identifier (UTI) for the edited image if the photo has been edited, otherwise None; for example: public.jpeg

**property uti\_original**

Returns Uniform Type Identifier (UTI) for the original image for example: public.jpeg or com.apple.quicktime-movie

**property uti\_raw**

Returns Uniform Type Identifier (UTI) for the RAW image if there is one for example: com.canon.cr2-raw-image Returns None if no associated RAW image

**property uuid**

UUID of picture

**property visible**

True if picture is visble

**property width**

returns width of the current photo version in pixels

## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`





## Symbols

- V
  - osxphotos-export command line option, 8
- album <ALBUM>
  - osxphotos-export command line option, 8
  - osxphotos-query command line option, 19
- album-keyword
  - osxphotos-export command line option, 13
- burst
  - osxphotos-export command line option, 9
  - osxphotos-query command line option, 20
- cleanup
  - osxphotos-export command line option, 15
- cloudasset
  - osxphotos-query command line option, 22
- convert-to-jpeg
  - osxphotos-export command line option, 12
- current-name
  - osxphotos-export command line option, 12
- db <Photos database path>
  - osxphotos command line option, 6
  - osxphotos-albums command line option, 6
  - osxphotos-dump command line option, 7
  - osxphotos-export command line option, 8
  - osxphotos-info command line option, 16
  - osxphotos-keywords command line option, 16
  - osxphotos-labels command line option, 17
  - osxphotos-persons command line option, 17
  - osxphotos-places command line option, 18
  - osxphotos-query command line option, 18
- deleted
  - osxphotos-dump command line option, 7
  - osxphotos-export command line option, 11
  - osxphotos-query command line option, 22
- deleted-only
  - osxphotos-dump command line option, 7
  - osxphotos-export command line option, 11
  - osxphotos-query command line option, 22
- description <DESC>
  - osxphotos-export command line option, 8
  - osxphotos-query command line option, 19
- description-template <TEMPLATE>
  - osxphotos-export command line option, 13
- directory <DIRECTORY>
  - osxphotos-export command line option, 14
- download-missing
  - osxphotos-export command line option, 12
- dry-run
  - osxphotos-export command line option, 11
- edited
  - osxphotos-export command line option, 9
  - osxphotos-query command line option, 17

```

    option, 19
--edited-suffix <SUFFIX>
    osxphotos-export command line
    option, 14
--exiftool
    osxphotos-export command line
    option, 13
--exiftool-merge-keywords
    osxphotos-export command line
    option, 13
--exiftool-merge-persons
    osxphotos-export command line
    option, 13
--exiftool-option <OPTION>
    osxphotos-export command line
    option, 13
--exiftool-path <EXIFTOOL_PATH>
    osxphotos-export command line
    option, 13
--export-as-hardlink
    osxphotos-export command line
    option, 11
--export-by-date
    osxphotos-export command line
    option, 12
--exportdb <EXPORTDB_FILE>
    osxphotos-export command line
    option, 15
--external-edit
    osxphotos-export command line
    option, 9
    osxphotos-query command line
    option, 19
--favorite
    osxphotos-export command line
    option, 9
    osxphotos-query command line
    option, 19
--filename <FILENAME>
    osxphotos-export command line
    option, 14
--finder-tag-keywords
    osxphotos-export command line
    option, 14
--finder-tag-template <TEMPLATE>
    osxphotos-export command line
    option, 13
--folder <FOLDER>
    osxphotos-export command line
    option, 8
    osxphotos-query command line
    option, 19
--from-date <from_date>
    osxphotos-export command line
    option, 10
    osxphotos-query command line
    option, 21
--from-time <from_time>
    osxphotos-export command line
    option, 10
    osxphotos-query command line
    option, 21
--has-comment
    osxphotos-export command line
    option, 10
    osxphotos-query command line
    option, 21
--has-likes
    osxphotos-export command line
    option, 10
    osxphotos-query command line
    option, 21
--has-raw
    osxphotos-export command line
    option, 10
    osxphotos-query command line
    option, 20
--hdr
    osxphotos-export command line
    option, 9
    osxphotos-query command line
    option, 20
--hidden
    osxphotos-export command line
    option, 9
    osxphotos-query command line
    option, 19
--ignore-case
    osxphotos-export command line
    option, 8
    osxphotos-query command line
    option, 19
--ignore-date-modified
    osxphotos-export command line
    option, 13
--ignore-signature
    osxphotos-export command line
    option, 11
--in-album
    osxphotos-export command line
    option, 10
    osxphotos-query command line
    option, 21
--includ
    osxphotos-query command line
    option, 22
--is-reference
    osxphotos-export command line

```

option,10	option,21
osxphotos-query command line	--min-size <SIZE>
option,21	osxphotos-export command line
--jpeg-ext <EXTENSION>	option,10
osxphotos-export command line	osxphotos-query command line
option,14	option,21
--jpeg-quality <jpeg_quality>	--missing
osxphotos-export command line	osxphotos-export command line
option,12	option,11
--json	osxphotos-query command line
osxphotos command line option,6	option,22
osxphotos-albums command line	--name <FILENAME>
option,6	osxphotos-export command line
osxphotos-dump command line option,	option,8
7	osxphotos-query command line
osxphotos-info command line option,	option,19
16	--no-comment
osxphotos-keywords command line	osxphotos-export command line
option,16	option,10
osxphotos-labels command line	osxphotos-query command line
option,17	option,21
osxphotos-list command line option,	--no-description
17	osxphotos-export command line
osxphotos-persons command line	option,8
option,17	osxphotos-query command line
osxphotos-places command line	option,19
option,18	--no-likes
osxphotos-query command line	osxphotos-export command line
option,18	option,10
--keyword <KEYWORD>	osxphotos-query command line
osxphotos-export command line	option,21
option,8	--no-place
osxphotos-query command line	osxphotos-export command line
option,18	option,8
--keyword-template <TEMPLATE>	osxphotos-query command line
osxphotos-export command line	option,19
option,13	--no-title
--label <LABEL>	osxphotos-export command line
osxphotos-export command line	option,8
option,8	osxphotos-query command line
osxphotos-query command line	option,19
option,19	--not-burst
--live	osxphotos-export command line
osxphotos-export command line	option,9
option,9	osxphotos-query command line
osxphotos-query command line	option,20
option,20	--not-cloudasset
--load-config <config file path>	osxphotos-query command line
osxphotos-export command line	option,22
option,15	--not-favorite
--max-size <SIZE>	osxphotos-export command line
osxphotos-export command line	option,9
option,11	osxphotos-query command line
osxphotos-query command line	option,19

```
--not-hdr
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,20
--not-hidden
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,19
--not-in-album
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,21
--not-includ
    osxphotos-query command line
        option,22
--not-live
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--not-missing
    osxphotos-query command line
        option,22
--not-panorama
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,20
--not-portrait
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--not-screenshot
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--not-selfie
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,20
--not-shared
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--not-slow-mo
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,21
--not-time-lapse
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--only-movies
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,21
--only-new
    osxphotos-export command line
        option,11
--only-photos
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,21
--original-suffix <SUFFIX>
    osxphotos-export command line
        option,14
--overwrite
    osxphotos-export command line
        option,11
--panorama
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,20
--person <PERSON>
    osxphotos-export command line
        option,8
    osxphotos-query command line
        option,18
--person-keyword
    osxphotos-export command line
        option,13
--place <PLACE>
    osxphotos-export command line
        option,8
    osxphotos-query command line
        option,19
--portrait
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--query-eval <CRITERIA>
    osxphotos-export command line
        option,11
    osxphotos-query command line
        option,21
```

---

```

--regex <REGEX TEMPLATE>
    osxphotos-export command line
        option,11
    osxphotos-query command line
        option,21
--replace-keywords
    osxphotos-export command line
        option,13
--report <path to export report>
    osxphotos-export command line
        option,14
--retry <RETRY>
    osxphotos-export command line
        option,11
--save-config <config file path>
    osxphotos-export command line
        option,15
--screenshot
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--selfie
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,20
--shared
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--sidecar <FORMAT>
    osxphotos-export command line
        option,12
--sidecar-drop-ext
    osxphotos-export command line
        option,12
--skip-bursts
    osxphotos-export command line
        option,12
--skip-edited
    osxphotos-export command line
        option,12
--skip-live
    osxphotos-export command line
        option,12
--skip-original-if-edited
    osxphotos-export command line
        option,12
--skip-raw
    osxphotos-export command line
        option,12
--slow-mo
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--strip
    osxphotos-export command line
        option,14
--time-lapse
    osxphotos-export command line
        option,9
    osxphotos-query command line
        option,20
--title <TITLE>
    osxphotos-export command line
        option,8
    osxphotos-query command line
        option,19
--to-date <to_date>
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,21
--to-time <to_time>
    osxphotos-export command line
        option,10
    osxphotos-query command line
        option,21
--touch-file
    osxphotos-export command line
        option,11
--update
    osxphotos-export command line
        option,11
--use-photokit
    osxphotos-export command line
        option,14
--use-photos-export
    osxphotos-export command line
        option,14
--uti <UTI>
    osxphotos-export command line
        option,8
    osxphotos-query command line
        option,19
--uuid <UUID>
    osxphotos-export command line
        option,8
    osxphotos-query command line
        option,19
--uuid-from-file <FILE>
    osxphotos-export command line
        option,8
    osxphotos-query command line
        option,19

```

```
--verbose
    osxphotos-export command line
        option, 8
--version
    osxphotos command line option, 6
--xattr-template <ATTRIBUTE TEMPLATE>
    osxphotos-export command line
        option, 14
-i
    osxphotos-export command line
        option, 8
    osxphotos-query command line
        option, 19
-v
    osxphotos command line option, 6
```

## A

activities() (*osxphotos.PhotoInfo.SearchInfo property*), 26

adjustments() (*osxphotos.PhotoInfo property*), 27

album\_info() (*osxphotos.PhotoInfo property*), 27

album\_info() (*osxphotos.PhotosDB property*), 22

album\_info\_shared() (*osxphotos.PhotosDB property*), 22

albums() (*osxphotos.PhotoInfo property*), 27

albums() (*osxphotos.PhotosDB property*), 22

albums\_as\_dict() (*osxphotos.PhotosDB property*), 22

albums\_shared() (*osxphotos.PhotosDB property*), 22

albums\_shared\_as\_dict() (*osxphotos.PhotosDB property*), 22

all() (*osxphotos.PhotoInfo.SearchInfo property*), 26

all\_files() (*osxphotos.PhotoInfo.ExportResults method*), 25

aperture (*osxphotos.PhotoInfo.ExifInfo attribute*), 24

asdict() (*osxphotos.PhotoInfo method*), 27

asdict() (*osxphotos.PhotoInfo.SearchInfo method*), 26

## B

behavioral (*osxphotos.PhotoInfo.ScoreInfo attribute*), 25

bit\_rate (*osxphotos.PhotoInfo.ExifInfo attribute*), 24

bodies\_of\_water() (*osxphotos.PhotoInfo.SearchInfo property*), 26

burst() (*osxphotos.PhotoInfo property*), 27

burst\_album\_info() (*osxphotos.PhotoInfo property*), 27

burst\_albums() (*osxphotos.PhotoInfo property*), 27

burst\_photos() (*osxphotos.PhotoInfo property*), 27

burst\_selected() (*osxphotos.PhotoInfo property*), 28

## C

camera\_make (*osxphotos.PhotoInfo.ExifInfo attribute*), 24

camera\_model (*osxphotos.PhotoInfo.ExifInfo attribute*), 24

city() (*osxphotos.PhotoInfo.SearchInfo property*), 26

codec (*osxphotos.PhotoInfo.ExifInfo attribute*), 25

comments() (*osxphotos.PhotoInfo property*), 28

country() (*osxphotos.PhotoInfo.SearchInfo property*), 26

curation (*osxphotos.PhotoInfo.ScoreInfo attribute*), 25

## D

date() (*osxphotos.PhotoInfo property*), 28

date\_modified() (*osxphotos.PhotoInfo property*), 28

date\_trashed() (*osxphotos.PhotoInfo property*), 28

db\_path() (*osxphotos.PhotosDB property*), 23

db\_version() (*osxphotos.PhotosDB property*), 23

description() (*osxphotos.PhotoInfo property*), 28

DEST

osxphotos-export command line option, 15

duration (*osxphotos.PhotoInfo.ExifInfo attribute*), 25

## E

exif\_info() (*osxphotos.PhotoInfo property*), 28

exiftool() (*osxphotos.PhotoInfo property*), 28

export() (*osxphotos.PhotoInfo method*), 28

export2() (*osxphotos.PhotoInfo method*), 29

exposure\_bias (*osxphotos.PhotoInfo.ExifInfo attribute*), 25

external\_edit() (*osxphotos.PhotoInfo property*), 30

## F

face\_info() (*osxphotos.PhotoInfo property*), 30

failure (*osxphotos.PhotoInfo.ScoreInfo attribute*), 25

favorite() (*osxphotos.PhotoInfo property*), 30

filename() (*osxphotos.PhotoInfo property*), 30

flash\_fired (*osxphotos.PhotoInfo.ExifInfo attribute*), 25

focal\_length (*osxphotos.PhotoInfo.ExifInfo attribute*), 25

folder\_info() (*osxphotos.PhotosDB property*), 23

folders() (*osxphotos.PhotosDB property*), 23

fps (*osxphotos.PhotoInfo.ExifInfo attribute*), 25

## G

get\_db\_connection() (*osxphotos.PhotosDB method*), 23

get\_photo() (*osxphotos.PhotosDB method*), 23

## H

harmonious\_color (*osxphotos.PhotoInfo.ScoreInfo* attribute), 25  
 has\_raw() (*osxphotos.PhotoInfo* property), 30  
 hasadjustments() (*osxphotos.PhotoInfo* property), 30  
 hdr() (*osxphotos.PhotoInfo* property), 31  
 height() (*osxphotos.PhotoInfo* property), 31  
 hidden() (*osxphotos.PhotoInfo* property), 31  
 highlight\_visibility (*osxphotos.PhotoInfo.ScoreInfo* attribute), 25  
 holidays() (*osxphotos.PhotoInfo.SearchInfo* property), 26

## I

immersiveness (*osxphotos.PhotoInfo.ScoreInfo* attribute), 25  
 import\_info() (*osxphotos.PhotoInfo* property), 31  
 import\_info() (*osxphotos.PhotosDB* property), 23  
 incloud() (*osxphotos.PhotoInfo* property), 31  
 interaction (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 interesting\_subject (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 intrash() (*osxphotos.PhotoInfo* property), 31  
 intrusive\_object\_presence (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 iscloudasset() (*osxphotos.PhotoInfo* property), 31  
 ismissing() (*osxphotos.PhotoInfo* property), 31  
 ismovie() (*osxphotos.PhotoInfo* property), 31  
 iso (*osxphotos.PhotoInfo.ExifInfo* attribute), 25  
 isphoto() (*osxphotos.PhotoInfo* property), 31  
 israw() (*osxphotos.PhotoInfo* property), 31  
 isreference() (*osxphotos.PhotoInfo* property), 31

## J

json() (*osxphotos.PhotoInfo* method), 31

## K

keywords() (*osxphotos.PhotoInfo* property), 31  
 keywords() (*osxphotos.PhotosDB* property), 23  
 keywords\_as\_dict() (*osxphotos.PhotosDB* property), 23

## L

labels() (*osxphotos.PhotoInfo* property), 31  
 labels() (*osxphotos.PhotoInfo.SearchInfo* property), 26  
 labels() (*osxphotos.PhotosDB* property), 23  
 labels\_as\_dict() (*osxphotos.PhotosDB* property), 23  
 labels\_normalized() (*osxphotos.PhotoInfo* property), 31

labels\_normalized() (*osxphotos.PhotosDB* property), 23  
 labels\_normalized\_as\_dict() (*osxphotos.PhotosDB* property), 23  
 latitude (*osxphotos.PhotoInfo.ExifInfo* attribute), 25  
 lens\_model (*osxphotos.PhotoInfo.ExifInfo* attribute), 25  
 library\_path() (*osxphotos.PhotosDB* property), 23  
 likes() (*osxphotos.PhotoInfo* property), 31  
 live\_photo() (*osxphotos.PhotoInfo* property), 31  
 lively\_color (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 locality\_names() (*osxphotos.PhotoInfo.SearchInfo* property), 27  
 location() (*osxphotos.PhotoInfo* property), 32  
 longitude (*osxphotos.PhotoInfo.ExifInfo* attribute), 25  
 low\_light (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26

## M

media\_types() (*osxphotos.PhotoInfo.SearchInfo* property), 27  
 metering\_mode (*osxphotos.PhotoInfo.ExifInfo* attribute), 25  
 month() (*osxphotos.PhotoInfo.SearchInfo* property), 27

## N

neighborhoods() (*osxphotos.PhotoInfo.SearchInfo* property), 27  
 noise (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26

## O

orientation() (*osxphotos.PhotoInfo* property), 32  
 original\_filename() (*osxphotos.PhotoInfo* property), 32  
 original\_filesize() (*osxphotos.PhotoInfo* property), 32  
 original\_height() (*osxphotos.PhotoInfo* property), 32  
 original\_orientation() (*osxphotos.PhotoInfo* property), 32  
 original\_width() (*osxphotos.PhotoInfo* property), 32  
 osxphotos command line option  
   --db <Photos database path>, 6  
   --json, 6  
   --version, 6  
   -v, 6  
 osxphotos-albums command line option  
   --db <Photos database path>, 6  
   --json, 6  
   PHOTOS\_LIBRARY, 7  
 osxphotos-dump command line option



```

--db <Photos database path>, 7
--deleted, 7
--deleted-only, 7
--json, 7
PHOTOS_LIBRARY, 7
osxphotos-export command line option
-V, 8
--album <ALBUM>, 8
--album-keyword, 13
--burst, 9
--cleanup, 15
--convert-to-jpeg, 12
--current-name, 12
--db <Photos database path>, 8
--deleted, 11
--deleted-only, 11
--description <DESC>, 8
--description-template <TEMPLATE>,
    13
--directory <DIRECTORY>, 14
--download-missing, 12
--dry-run, 11
--edited, 9
--edited-suffix <SUFFIX>, 14
--exiftool, 13
--exiftool-merge-keywords, 13
--exiftool-merge-persons, 13
--exiftool-option <OPTION>, 13
--exiftool-path <EXIFTOOL_PATH>, 13
--export-as-hardlink, 11
--export-by-date, 12
--exportdb <EXPORTDB_FILE>, 15
--external-edit, 9
--favorite, 9
--filename <FILENAME>, 14
--finder-tag-keywords, 14
--finder-tag-template <TEMPLATE>, 13
--folder <FOLDER>, 8
--from-date <from_date>, 10
--from-time <from_time>, 10
--has-comment, 10
--has-likes, 10
--has-raw, 10
--hdr, 9
--hidden, 9
--ignore-case, 8
--ignore-date-modified, 13
--ignore-signature, 11
--in-album, 10
--is-reference, 10
--jpeg-ext <EXTENSION>, 14
--jpeg-quality <jpeg_quality>, 12
--keyword <KEYWORD>, 8
--keyword-template <TEMPLATE>, 13
--label <LABEL>, 8
--live, 9
--load-config <config file path>, 15
--max-size <SIZE>, 11
--min-size <SIZE>, 10
--missing, 11
--name <FILENAME>, 8
--no-comment, 10
--no-description, 8
--no-likes, 10
--no-place, 8
--no-title, 8
--not-burst, 9
--not-favorite, 9
--not-hdr, 10
--not-hidden, 9
--not-in-album, 10
--not-live, 9
--not-panorama, 10
--not-portrait, 9
--not-screenshot, 9
--not-selfie, 10
--not-shared, 9
--not-slow-mo, 9
--not-time-lapse, 9
--only-movies, 10
--only-new, 11
--only-photos, 10
--original-suffix <SUFFIX>, 14
--overwrite, 11
--panorama, 10
--person <PERSON>, 8
--person-keyword, 13
--place <PLACE>, 8
--portrait, 9
--query-eval <CRITERIA>, 11
--regex <REGEX_TEMPLATE>, 11
--replace-keywords, 13
--report <path to export report>, 14
--retry <RETRY>, 11
--save-config <config file path>, 15
--screenshot, 9
--selfie, 10
--shared, 9
--sidecar <FORMAT>, 12
--sidecar-drop-ext, 12
--skip-bursts, 12
--skip-edited, 12
--skip-live, 12
--skip-original-if-edited, 12
--skip-raw, 12
--slow-mo, 9
--strip, 14
--time-lapse, 9

```



```

--title <TITLE>, 8
--to-date <to_date>, 10
--to-time <to_time>, 10
--touch-file, 11
--update, 11
--use-photokit, 14
--use-photos-export, 14
--uti <UTI>, 8
--uuid <UUID>, 8
--uuid-from-file <FILE>, 8
--verbose, 8
--xattr-template <ATTRIBUTE
    TEMPLATE>, 14
-i, 8
DEST, 15
PHOTOS_LIBRARY, 15
osxphotos-help command line option
    TOPIC, 15
osxphotos-info command line option
    --db <Photos database path>, 16
    --json, 16
    PHOTOS_LIBRARY, 16
osxphotos-keywords command line option
    --db <Photos database path>, 16
    --json, 16
    PHOTOS_LIBRARY, 16
osxphotos-labels command line option
    --db <Photos database path>, 17
    --json, 17
    PHOTOS_LIBRARY, 17
osxphotos-list command line option
    --json, 17
osxphotos-persons command line option
    --db <Photos database path>, 17
    --json, 17
    PHOTOS_LIBRARY, 18
osxphotos-places command line option
    --db <Photos database path>, 18
    --json, 18
    PHOTOS_LIBRARY, 18
osxphotos-query command line option
    --album <ALBUM>, 19
    --burst, 20
    --cloudasset, 22
    --db <Photos database path>, 18
    --deleted, 22
    --deleted-only, 22
    --description <DESC>, 19
    --edited, 19
    --external-edit, 19
    --favorite, 19
    --folder <FOLDER>, 19
    --from-date <from_date>, 21
    --from-time <from_time>, 21
    --has-comment, 21
    --has-likes, 21
    --has-raw, 20
    --hdr, 20
    --hidden, 19
    --ignore-case, 19
    --in-album, 21
    --incloud, 22
    --is-reference, 21
    --json, 18
    --keyword <KEYWORD>, 18
    --label <LABEL>, 19
    --live, 20
    --max-size <SIZE>, 21
    --min-size <SIZE>, 21
    --missing, 22
    --name <FILENAME>, 19
    --no-comment, 21
    --no-description, 19
    --no-likes, 21
    --no-place, 19
    --no-title, 19
    --not-burst, 20
    --not-cloudasset, 22
    --not-favorite, 19
    --not-hdr, 20
    --not-hidden, 19
    --not-in-album, 21
    --not-incloud, 22
    --not-live, 20
    --not-missing, 22
    --not-panorama, 20
    --not-portrait, 20
    --not-screenshot, 20
    --not-selfie, 20
    --not-shared, 20
    --not-slow-mo, 20
    --not-time-lapse, 20
    --only-movies, 21
    --only-photos, 21
    --panorama, 20
    --person <PERSON>, 18
    --place <PLACE>, 19
    --portrait, 20
    --query-eval <CRITERIA>, 21
    --regex <REGEX TEMPLATE>, 21
    --screenshot, 20
    --selfie, 20
    --shared, 20
    --slow-mo, 20
    --time-lapse, 20
    --title <TITLE>, 19
    --to-date <to_date>, 21
    --to-time <to_time>, 21

```

--uti <UTI>, 19  
 --uuid <UUID>, 19  
 --uuid-from-file <FILE>, 19  
 -i, 19  
 PHOTOS\_LIBRARY, 22  
 overall (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26

## P

panorama() (*osxphotos.PhotoInfo* property), 32  
 path() (*osxphotos.PhotoInfo* property), 32  
 path\_edited() (*osxphotos.PhotoInfo* property), 32  
 path\_live\_photo() (*osxphotos.PhotoInfo* property), 32  
 path\_raw() (*osxphotos.PhotoInfo* property), 32  
 person\_info() (*osxphotos.PhotoInfo* property), 32  
 person\_info() (*osxphotos.PhotosDB* property), 23  
 persons() (*osxphotos.PhotoInfo* property), 32  
 persons() (*osxphotos.PhotosDB* property), 23  
 persons\_as\_dict() (*osxphotos.PhotosDB* property), 23  
 PhotoInfo (class in *osxphotos*), 24  
 PhotoInfo.ExifInfo (class in *osxphotos*), 24  
 PhotoInfo.ExportResults (class in *osxphotos*), 25  
 PhotoInfo.ScoreInfo (class in *osxphotos*), 25  
 PhotoInfo.SearchInfo (class in *osxphotos*), 26  
 photos() (*osxphotos.PhotosDB* method), 23  
 photos\_by\_uuid() (*osxphotos.PhotosDB* method), 24  
 PHOTOS\_LIBRARY  
   osxphotos-albums command line option, 7  
   osxphotos-dump command line option, 7  
   osxphotos-export command line option, 15  
   osxphotos-info command line option, 16  
   osxphotos-keywords command line option, 16  
   osxphotos-labels command line option, 17  
   osxphotos-persons command line option, 18  
   osxphotos-places command line option, 18  
   osxphotos-query command line option, 22  
 PhotosDB (class in *osxphotos*), 22  
 place() (*osxphotos.PhotoInfo* property), 32  
 place\_names() (*osxphotos.PhotoInfo.SearchInfo* property), 27  
 pleasant\_camera\_tilt (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26

pleasant\_composition (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 pleasant\_lighting (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 pleasant\_pattern (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 pleasant\_perspective (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 pleasant\_post\_processing (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 pleasant\_reflection (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 pleasant\_symmetry (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 portrait() (*osxphotos.PhotoInfo* property), 32  
 promotion (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26

## Q

query() (*osxphotos.PhotosDB* method), 24

## R

raw\_original() (*osxphotos.PhotoInfo* property), 32  
 render\_template() (*osxphotos.PhotoInfo* method), 32

## S

sample\_rate (*osxphotos.PhotoInfo.ExifInfo* attribute), 25  
 score() (*osxphotos.PhotoInfo* property), 33  
 screenshot() (*osxphotos.PhotoInfo* property), 33  
 search\_info() (*osxphotos.PhotoInfo* property), 33  
 search\_info\_normalized() (*osxphotos.PhotoInfo* property), 33  
 season() (*osxphotos.PhotoInfo.SearchInfo* property), 27  
 selfie() (*osxphotos.PhotoInfo* property), 33  
 shared() (*osxphotos.PhotoInfo* property), 33  
 sharply\_focused\_subject (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 shutter\_speed (*osxphotos.PhotoInfo.ExifInfo* attribute), 25  
 slow\_mo() (*osxphotos.PhotoInfo* property), 33  
 state() (*osxphotos.PhotoInfo.SearchInfo* property), 27  
 state\_abbreviation() (*osxphotos.PhotoInfo.SearchInfo* property), 27  
 streets() (*osxphotos.PhotoInfo.SearchInfo* property), 27

## T

tastefully\_blurred (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26  
 time\_lapse() (*osxphotos.PhotoInfo* property), 33  
 title() (*osxphotos.PhotoInfo* property), 33

## TOPIC

osxphotos-help command line option,  
15

track\_format (*osxphotos.PhotoInfo.ExifInfo* attribute), 25

tzoffset() (*osxphotos.PhotoInfo* property), 33

## U

uti() (*osxphotos.PhotoInfo* property), 33

uti\_edited() (*osxphotos.PhotoInfo* property), 33

uti\_original() (*osxphotos.PhotoInfo* property), 33

uti\_raw() (*osxphotos.PhotoInfo* property), 33

uuid() (*osxphotos.PhotoInfo* property), 34

## V

venue\_types() (*osxphotos.PhotoInfo.SearchInfo* property), 27

venues() (*osxphotos.PhotoInfo.SearchInfo* property),  
27

visible() (*osxphotos.PhotoInfo* property), 34

## W

well\_chosen\_subject (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26

well\_framed\_subject (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26

well\_timed\_shot (*osxphotos.PhotoInfo.ScoreInfo* attribute), 26

white\_balance (*osxphotos.PhotoInfo.ExifInfo* attribute), 25

width() (*osxphotos.PhotoInfo* property), 34

## Y

year() (*osxphotos.PhotoInfo.SearchInfo* property), 27