
PhotoScript

Release 0.0.12

Rhet Turnbull

Sep 09, 2020

CONTENTS

1	PhotoScript	1
1.1	What is PhotoScript	1
1.2	Installation	1
1.3	Example	1
1.4	See Also	2
2	Documentation	3
2.1	photoscript package	3
2.1.1	photoscript module	3
3	Indices and tables	9
	Index	11

PHOTOSCRIPT

1.1 What is PhotoScript

PhotoScript provides a python wrapper around Apple Photos applescript interface. With PhotoScript you can interact with Photos using python. Runs only on MacOS. Tested on MacOS Catalina.

1.2 Installation

PhotoScript uses setuptools, thus simply run:

```
python3 setup.py install
```

1.3 Example

```
""" Simple example showing use of photoscript """

import photoscript

photoslib = photoscript.PhotosLibrary()

photoslib.activate()
print(f"Running Photos version: {photoslib.version}")

album = photoslib.album("Album1")
photos = album.photos

for photo in photos:
    print(f"{photo.title}, {photo.description}, {photo.keywords}")

new_album = photoslib.create_album("New Album")
photoslib.import_photos(["/Users/rhet/Downloads/test.jpeg"], album=new_album)

photoslib.quit()
```

1.4 See Also

- [osxphotos](#): Python package that provides read-only access to the Photos library including all associated meta-data.

DOCUMENTATION

2.1 photoscript package

2.1.1 photoscript module

class `photoscript.PhotosLibrary`

activate ()

activate Photos.app

album (**name*, *uuid=None*, *top_level=False*)

Album instance by name or id

Parameters

- **name** – name of album
- **uuid** – id of album
- **top_level** – if True, searches only top level albums; default = False

Returns Album object or None if album could not be found

Raises **ValueError** if both name and id passed. –

Must pass only name or id but not both. If more than one album with same name, returns the first one found.

album_names (*top_level=False*)

List of album names in the Photos library

Parameters **top_level** – if True, returns only top-level albums otherwise also returns albums in sub-folders; default is False

albums (*top_level=False*)

list of Album objects for all albums

create_album (*name*, *folder=None*)

creates an album

Parameters

- **name** – name of new album
- **folder** – Folder object in which to create new album. If None, creates top-level album. Default is None.

Returns Album object for newly created album

Raises `AppleScriptError` if error creating the album –

`create_folder` (*name*, *folder=None*)
creates a folder

Parameters

- **name** – name of new folder
- **folder** – Folder object in which to create the new folder. If None, creates top-level folder. Default is None.

Returns Folder object for newly created folder

Raises `AppleScriptError` if folder cannot be created –

`delete_album` (*album*)
deletes album (but does not delete photos in the album)

Parameters **album** – an Album object for album to delete

`delete_folder` (*folder*)
Deletes folder (Not yet implemented)

Parameters **folder** – a Folder object for folder to delete

`property favorites`
Album object for the Favorites album

`folder` (**name*, *uuid=None*, *top_level=True*)
Folder instance by name or uuid

Parameters

- **name** – name of folder
- **uuid** – id of folder
- **top_level** – if True, only searches top level folders by name; default is True

Returns Folder object or None if folder could not be found

Raises `ValueError` if both name and id passed. –

Must pass only name or id but not both. If more than one folder with same name, returns first one found.

`folder_by_path` (*folder_path*)
Return folder in the library by path

Parameters **folder_path** – list of folder names in descending path order, e.g. ["Folder", "SubFolder1", "SubFolder2"]

Returns Folder object for folder at folder_path or None if not found

`folder_names` (*top_level=False*)
List of folder names in the Photos library

Parameters **top_level** – if True, returns only top-level folders otherwise also returns sub-folders; default is False

`folders` (*top_level=True*)
list of Folder objects for all folders

`property frontmost`
True if Photos.app is front most app otherwise False

import_photos (*photo_paths*, *album=None*, *skip_duplicate_check=False*)
 import photos

Parameters

- **photo_paths** – list of file paths to import as str or pathlib.Path
- **album** – optional, Album object for album to import into
- **skip_duplicate_check** – if True, Photos will not check for duplicates on import, default is False.
- **NOTE** – If you attempt to import a duplicate photo and `skip_duplicate_check != True`,
- **will block with drop-down sheet until the user clicks "Cancel (Photos) –**
- **Import –**
- **Don't Import." (or) –**

Returns list of Photo objects for imported photos

make_album_folders (*album_name*, *folder_path*)

Make album in a folder path. If either the album or any component of the folder path doesn't exist, it will be created. If album or folder path does exist, no duplicate is created. Folder path is created recursively if needed.

Parameters

- **album_name** – name of album to create. If album already exists, returns existing album.
- **folder_path** – list of folder names in descending path order, e.g. ["Folder", "SubFolder1", "SubFolder2"].

Returns Album object.

Raises

- **ValueError** if *folder_path* is empty or *album_name* is None. –
- **TypeError** if *folder_path* is not a list. –

make_folders (*folder_path*)

Recursively makes folders and subfolders. Works similar to `os.makedirs`. If any component of *folder_path* already exists, does not raise error.

Parameters *folder_path* – list of folder names in descending path order, e.g. ["Folder", "SubFolder1", "SubFolder2"]

Returns Folder object for the final sub folder

Raises

- **ValueError** if *folder_path* is empty –
- **TypeError** if *folder_path* is not a list –

property name

name of Photos.app

photos (*search=None*, *uuid=None*)

List of Photo objects for items in the library Note: for a large library, calling `photos()` may run a *very* long time (minutes)

Parameters

- **search** – optional text string to search for (returns matching items)
- **uuid** – list of UUIDs to get
- **may pass search or uuid but not both** (*you*) –

Returns List of Photo objects or [] if no photos found

Raises ValueError if both search and uuid are passed –

quit ()

quit Photos.app

property selection

List of Photo objects for currently selected photos or [] if no selection

property version

version of Photos.app as str

class photoscript.**Album**(*uuid*)

add(*photos*)

add photos from the library to album

Parameters **photos** – list of Photo objects to add to album

export (*path, original=True, edited=True, timeout=120*)

Export photos in album to path

Parameters

- **path** – path to export to
- **original** – if True exports original photo
- **edited** – if True, exports edited version, if one exists
- **timeout** – number of seconds to timeout waiting for Photos to respond

Returns list of names of exported photos

import_photos (*photo_paths, skip_duplicate_check=False*)

import photos

Parameters

- **photos** – list of file paths to import
- **skip_duplicate_check** – if True, Photos will not check for duplicates on import, default is False

Returns list of Photo objects for impoted photos

property name

name of album

property parent

Return parent Folder object

property parent_id

parent container id

path_str (*delim='/'*)

Return internal library path to album as string. e.g. “Folder/SubFolder/AlbumName”

Parameters `delim` – character to use as delimiter between path elements; default is “/”

Raises `ValueError` if `delim` is not a single character –

photos ()

list of Photo objects for photos contained in album

remove (*photos*)

Remove photos from album. Note: Photos does not provide a way to remove photos from an album via AppleScript. This method actually creates a new Album with the same name as the original album and copies all photos from original album with exception of those to remove to the new album then deletes the old album.

Parameters `photos` – list of Photo objects to remove

Returns new Album object for the new album with photos removed.

remove_by_id (*photo_ids*)

Remove photos from album. Note: Photos does not provide a way to remove photos from an album via AppleScript. This method actually creates a new Album with the same name as the original album and copies all photos from original album with exception of those to remove to the new album then deletes the old album.

Parameters `photo_ids` – list of photo ids to remove

Returns new Album object for the new album with photos removed.

property `title`

title of album (alias for `Album.name`)

property `uuid`

UUID of Album

class `photoscript.Photo` (*uuid*)

property `altitude`

GPS altitude of photo in meters

property `date`

date of photo as timezone-naive `datetime.datetime` object

property `description`

description of photo

duplicate ()

duplicates the photo and returns Photo object for the duplicate

export (*path*, *original=True*, *edited=True*, *timeout=120*)

Export photo

Parameters

- **path** – path to export to
- **original** – if True exports original photo
- **edited** – if True, exports edited version, if one exists
- **timeout** – number of seconds to timeout waiting for Photos to respond

Returns name of exported photo

property favorite

return favorite status (boolean)

property filename

filename of photo

property height

height of photo in pixels

property keywords

list of keywords for photo

property location

The GPS latitude and longitude, in a tuple of 2 numbers or None. Latitude in range -90.0 to 90.0, longitude in range -180.0 to 180.0.

property name

name of photo

property title

title of photo (alias for name)

property uuid

UUID of Photo

property width

width of photo in pixels

INDICES AND TABLES

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

A

activate() (*photoscript.PhotosLibrary method*), 3
 add() (*photoscript.Album method*), 6
 Album (*class in photoscript*), 6
 album() (*photoscript.PhotosLibrary method*), 3
 album_names() (*photoscript.PhotosLibrary method*), 3
 albums() (*photoscript.PhotosLibrary method*), 3
 altitude() (*photoscript.Photo property*), 7

C

create_album() (*photoscript.PhotosLibrary method*), 3
 create_folder() (*photoscript.PhotosLibrary method*), 4

D

date() (*photoscript.Photo property*), 7
 delete_album() (*photoscript.PhotosLibrary method*), 4
 delete_folder() (*photoscript.PhotosLibrary method*), 4
 description() (*photoscript.Photo property*), 7
 duplicate() (*photoscript.Photo method*), 7

E

export() (*photoscript.Album method*), 6
 export() (*photoscript.Photo method*), 7

F

favorite() (*photoscript.Photo property*), 8
 favorites() (*photoscript.PhotosLibrary property*), 4
 filename() (*photoscript.Photo property*), 8
 folder() (*photoscript.PhotosLibrary method*), 4
 folder_by_path() (*photoscript.PhotosLibrary method*), 4
 folder_names() (*photoscript.PhotosLibrary method*), 4
 folders() (*photoscript.PhotosLibrary method*), 4
 frontmost() (*photoscript.PhotosLibrary property*), 4

H

height() (*photoscript.Photo property*), 8

I

import_photos() (*photoscript.Album method*), 6
 import_photos() (*photoscript.PhotosLibrary method*), 4

K

keywords() (*photoscript.Photo property*), 8

L

location() (*photoscript.Photo property*), 8

M

make_album_folders() (*photoscript.PhotosLibrary method*), 5
 make_folders() (*photoscript.PhotosLibrary method*), 5

N

name() (*photoscript.Album property*), 6
 name() (*photoscript.Photo property*), 8
 name() (*photoscript.PhotosLibrary property*), 5

P

parent() (*photoscript.Album property*), 6
 parent_id() (*photoscript.Album property*), 6
 path_str() (*photoscript.Album method*), 6
 Photo (*class in photoscript*), 7
 photos() (*photoscript.Album method*), 7
 photos() (*photoscript.PhotosLibrary method*), 5
 PhotosLibrary (*class in photoscript*), 3

Q

quit() (*photoscript.PhotosLibrary method*), 6

R

remove() (*photoscript.Album method*), 7
 remove_by_id() (*photoscript.Album method*), 7

S

`selection()` (*photoscript.PhotosLibrary property*), [6](#)

T

`title()` (*photoscript.Album property*), [7](#)

`title()` (*photoscript.Photo property*), [8](#)

U

`uuid()` (*photoscript.Album property*), [7](#)

`uuid()` (*photoscript.Photo property*), [8](#)

V

`version()` (*photoscript.PhotosLibrary property*), [6](#)

W

`width()` (*photoscript.Photo property*), [8](#)