

---

# **PhotoScript**

***Release 0.0.20***

**Rhet Turnbull**

**Sep 14, 2020**



# CONTENTS

<b>1</b>	<b>PhotoScript</b>	<b>1</b>
1.1	What is PhotoScript . . . . .	1
1.2	Installation . . . . .	1
1.3	Example . . . . .	1
1.4	See Also . . . . .	2
<b>2</b>	<b>Documentation</b>	<b>3</b>
2.1	photoscript package . . . . .	3
2.1.1	photoscript module . . . . .	3
<b>3</b>	<b>Indices and tables</b>	<b>11</b>
	<b>Index</b>	<b>13</b>



## 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 or neither 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

**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 or neither 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



**property hidden**

True if Photos is hidden (or not running), False if Photos is visible

**hide()**

Tell Photos to hide its window

**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.

**Returns** list of Photo objects for imported photos

**NOTE: If you attempt to import a duplicate photo and `skip_duplicate_check != True`, Photos will block with drop-down sheet until the user clicks “Cancel, Import, or Don’t Import.”**

**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 running**

True if Photos is running, otherwise False

**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

**Returns** list of Photo objects for added photos

**export** (*export\_path, original=False, overwrite=False, timeout=120*)

Export photos in album to path

**Parameters**

- **photo** – Photo object to export
- **export\_path** – path to export to
- **original** – if True, export original image, otherwise export current image; default = False
- **overwrite** – if True, export will overwrite a file of same name as photo in export\_path; default = False
- **timeout** – number of seconds to wait for Photos to complete export (for each photo) before timing out; default = 120

**Returns** List of full paths of exported photos. There may be more than one photo exported due to live images and burst images.

**Raises ValueError if export\_path is not a valid directory –**

Note: Photos always exports as high-quality JPEG unless original=True. If original=True, will export all burst images for burst photos and live movie for live photos. If original=False, only the primary image from a burst set will be exported for burst photos and the live movie component of a live image will not be exported, only the JPEG component.

**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 imported photos

**property name**  
 name of album (read/write)

**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 (read only)

**class** **photoscript.Folder** (*uuid*)

**album** (*name*)

Return Album object contained in this folder for album named *name* or None if no matching album

**property albums**

list of Album objects for albums contained in folder

**create\_album** (*name*)

Creates an album in this folder

**Parameters** *name* – name of new album

**Returns** Album object for newly created album

**create\_folder** (*name*)

creates a folder in this folder

**Returns** Folder object for newly created folder

**folder** (*name*)

Folder object for first subfolder folder named *name*.

**Parameters** *name* – name of folder to return

**Returns** Folder object for first subfolder who's name matches *name* or None if not found

**property name**

name of folder

**property parent**

Return parent Folder object

**property parent\_id**

parent container id

**path** ()

Return list of Folder objects this folder is contained in. `path()[0]` is the top-level folder this folder is contained in and `path()[-1]` is the immediate parent of this folder. Returns empty list if folder is not contained in another folders.

**path\_str** (*delim*='/')

**Return internal library path to folder as string.** e.g. "Folder/SubFolder"

**Parameters** *delim* – character to use as delimiter between path elements; default is "/"

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

**property subfolders**

list of Folder objects for immediate sub-folders contained in folder

**property title**

title of folder (alias for `Folder.name`)

**property uuid**

UUID of folder

**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** (*export\_path*, *original=False*, *overwrite=False*, *timeout=120*)

Export photo

**Parameters**

- **photo** – Photo object to export
- **export\_path** – path to export to
- **original** – if True, export original image, otherwise export current image; default = False
- **overwrite** – if True, export will overwrite a file of same name as photo in export\_path; default = False
- **timeout** – number of seconds to wait for Photos to complete export before timing out; default = 120

**Returns** List of full paths of exported photos. There may be more than one photo exported due to live images and burst images.

**Raises ValueError if export\_path is not a valid directory –**

Note: Photos always exports as high-quality JPEG unless original=True. If original=True, will export all burst images for burst photos and live movie for live photos. If original=False, only the primary image from a burst set will be exported for burst photos and the live movie component of a live image will not be exported, only the JPEG component.

**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.Folder method*), 7  
 album() (*photoscript.PhotosLibrary method*), 3  
 album\_names() (*photoscript.PhotosLibrary method*), 3  
 albums() (*photoscript.Folder property*), 8  
 albums() (*photoscript.PhotosLibrary method*), 3  
 altitude() (*photoscript.Photo property*), 8

## C

create\_album() (*photoscript.Folder method*), 8  
 create\_album() (*photoscript.PhotosLibrary method*), 3  
 create\_folder() (*photoscript.Folder method*), 8  
 create\_folder() (*photoscript.PhotosLibrary method*), 4

## D

date() (*photoscript.Photo property*), 8  
 delete\_album() (*photoscript.PhotosLibrary method*), 4  
 delete\_folder() (*photoscript.PhotosLibrary method*), 4  
 description() (*photoscript.Photo property*), 8  
 duplicate() (*photoscript.Photo method*), 9

## E

export() (*photoscript.Album method*), 6  
 export() (*photoscript.Photo method*), 9

## F

favorite() (*photoscript.Photo property*), 9  
 favorites() (*photoscript.PhotosLibrary property*), 4  
 filename() (*photoscript.Photo property*), 9  
 Folder (*class in photoscript*), 7  
 folder() (*photoscript.Folder method*), 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*), 9  
 hidden() (*photoscript.PhotosLibrary property*), 4  
 hide() (*photoscript.PhotosLibrary method*), 5

## I

import\_photos() (*photoscript.Album method*), 6  
 import\_photos() (*photoscript.PhotosLibrary method*), 5

## K

keywords() (*photoscript.Photo property*), 9

## L

location() (*photoscript.Photo property*), 9

## M

make\_album\_folders() (*photoscript.PhotosLibrary method*), 5  
 make\_folders() (*photoscript.PhotosLibrary method*), 5

## N

name() (*photoscript.Album property*), 7  
 name() (*photoscript.Folder property*), 8  
 name() (*photoscript.Photo property*), 9  
 name() (*photoscript.PhotosLibrary property*), 5

## P

parent() (*photoscript.Album property*), 7  
 parent() (*photoscript.Folder property*), 8  
 parent\_id() (*photoscript.Album property*), 7  
 parent\_id() (*photoscript.Folder property*), 8  
 path() (*photoscript.Folder method*), 8  
 path\_str() (*photoscript.Album method*), 7  
 path\_str() (*photoscript.Folder method*), 8

`Photo` (*class in photoscript*), 8  
`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  
`running()` (*photoscript.PhotosLibrary property*), 6

## S

`selection()` (*photoscript.PhotosLibrary property*), 6  
`subfolders()` (*photoscript.Folder property*), 8

## T

`title()` (*photoscript.Album property*), 7  
`title()` (*photoscript.Folder property*), 8  
`title()` (*photoscript.Photo property*), 9

## U

`uuid()` (*photoscript.Album property*), 7  
`uuid()` (*photoscript.Folder property*), 8  
`uuid()` (*photoscript.Photo property*), 9

## V

`version()` (*photoscript.PhotosLibrary property*), 6

## W

`width()` (*photoscript.Photo property*), 9