PhotoScript

Release 0.0.24

Rhet Turnbull

Sep 19, 2020

CONTENTS

CHAPTER

ONE

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 metadata.

CHAPTER

TWO

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 impoted 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*, *range_=None*)

Returns a generator that yields Photo objects for media items in the library.

Parameters

- **search** – optional text string to search for (returns matching items)
- **uuid** – optional list of UUIDs to get
- **range_** – optional list of [start, stop] sequence of photos to get

Returns Generator that yields Photo objects

Raises

- **ValueError if more than one of search, uuid, range_ passed or invalid range_** –
- **TypeError if list not passed for range_** –

Note: photos() returns a generator instead of a list because retrieving all the photos from a large Photos library can take a very long time–on my system, the rate is about 1 per second; this is limited by the Photos AppleScript interface and I've not found anyway to speed it up. Using a generator allows you process photos individually rather than waiting, possibly hours, for Photos to return the results.

range_ works like python's range function. Thus range_=[0,4] will return Photos 0, 1, 2, 3; range_=[10] returns the first 10 photos in the library; range_ start must be in range 0 to len(PhotosLibrary())-1, stop in range 1 to len(PhotosLibrary()). You may be able to optimize the speed by which photos are return by chunking up requests in batches of photos using range_, e.g. request 10 photos at a time.

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*, *reveal_in_finder=False*)
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
- **reveal_in_finder** – if True, will open Finder with exported items selected when done; default = False

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 impoted 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.

spotlight()
spotlight the album in Photos

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 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** –

spotlight()
spotlight the folder in Photos

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*, *reveal_in_finder=False*)
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
- **reveal_in_finder** – if True, will open Finder with exported items selected when done; default = False

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

spotlight()
spotlight the photo in Photos

property title
title of photo (alias for name)

property uuid
UUID of Photo

property width
width of photo in pixels

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

INDEX

Q

R

S

T

U

V

W