photorename.py

A Dedicated Cross-Platform Image Renamer

Gene Wilburn, 2024

photorename.py is NOT a generalized photo and image renamer. It is command-line Python3 utility designed to batch rename a set of images in a specified working directory to the following format:

```
yyyymmdd-sequencenumber+photographer--usertags.file-extension
```

e.g.:

```
20240610-015g--port-credit-lighthouse.jpg
```

If called by itself, `photorename.py` creates a date format filename with a "tags" placeholder left open for further renaming. This is the principle way of invoking the script when you have a mixed directory of images and need to add separate tags to each image or to subsets of images. e.g.:

```
20240610-021g--tags.jpg
```

As you view these images in the image viewer of your choice, you can add specific tags through the image viewer's rename function. Popular viewers that allow you to further edit these image names include Adobe Bridge (Mac/Windows), Irfanview (Windows), Preview (Mac), and eom (Eye of Mate) for Linux.

`photorename.py` takes two arguments that can be optionally be passed in to the program:

- $1 is the photographer's initial(s), especially if it's a different photographer from you.
- $2 is a user-supplied set of tags, hyphen-delimited.

When I process my wife's photos I use 'm' (Marion) for the intial and if I add a set of tags, a call to the script look this:

```
photorename.py m art-in-the-park-port-credit
```

This batch of photos would be rendered as:

```
20240610-007m--art-in-the-park-port-credit.jpg
```

The script automatically provides sequence numbers of photos taken the same day, based on embedded EXIF 'DateTime' data. When the day changes, the sequence number is set back to '1' and restarted.

Philosophy

`photorename.py` embraces the philosophy that 'since filenames can be up to at least 250 characters long, use that to your advantage.' Put in all the tags you need into the filename itself for easy retrieval and to keep things open and visible. This open approach allows you to use any file searching tool to fetch the images you might be looking for.

In most cases your preferred operating system can search filenames and display the image as thumbnails. Or, as I have, you can create a search script that gathers all the eligible hits into an array and pass them to a good photo viewer to examine each one enlarged.

Requirements

`photorename.py`, as mentioned, is a Python3 script that you can run in the operating system of your choice. There are two requirements:

- a recent release of Python3
- the 'Pillow' (PIL—Python Image Library) module

Installing Python3 and Pillow on your operating system

Linux

Python3 comes preinstalled in all major Linux distros. All you need to do is install the 'Pillow' module. In Ubuntu-based systems this is easily done by typing:

```
sudo apt install python3-pillow
```

In Fedora, CentOS, RHEL type:

```
sudo dnf install python3-pillow
```

MacOS

For MacOS you must have the Command Line Utilities from Apple installed. Look up 'Command Line Utilities' in Mac's App Store and install them if you haven't already.

Once that's installed, I further suggest you install Homebrew (follow the instructions on the Homebrew website). Homebrew offers the latest in open-source command-line utilities for your Mac. Once that's installed, type:

```
brew install pillow
```

and you'll be all set to use the `photorename.py` script.

Windows

Windows doesn't have Python3 installed by default. To install Python3, go to python.org and in the "Downloads" section select the latest stable version for Windows. Download the installer and double click on it to run it. Check the box that says "Add Python to PATH", then click "Install Now."

To verify that it installed correctly, press the Windows key and type 'cmd' and press Enter. Then type 'python --version' (without the quotes). If it's installed okay it will display the version number. On the next line type 'pip install Pillow':

```
C:\Users\gene\python \-\-version
Python 3.12.6
```

C:\Users\gene\pip install Pillow`

Where To Store `photorename.py`

Linux

The most obvious place to put `photorename.py` is in your $HOME/bin directory. If you don't have a bin/ directory you can make one by starting a terminal and typing 'mkdir bin'. Many .bashrc scripts are already set up to add $HOME/bin to your pathname if it exists. Otherwise, edit .bashrc and on the last line of the script type:

```
export PATH="$HOME/bin;$PATH"
```

Then type

`source .bashrc` to update your current session to the new path.

If you're using Z shell rather than Bash, do the same for .zshrc.

MacOS

To keep in line with Linux, I suggest you follow the Linux practice of creating a `bin` directory under your username home directory and adding the extra path statement to either .bashrc or .zshrc, whichever you are using, as in the instructions for Linux. In recent years, MacOS has preferred Z shell. You can edit either the .bashrc file or the .zshrc file with Mac's TextEdit program if you don't already have a preferred text editor.

Windows

In your home directory you could create a folder with Windows Explorer called `MyUtilties` or something similar, but over time it may be less confusing just to create a folder called `bin` as it is usually named on Linux or Mac. The name `bin` is short for `binaries` and is frequently used as a place to store executable programs such as the `photorename.py` script.

Windows, alas, is always a bit more convoluted than it needs to be, but you now want your `bin` directory in your system's Pathfile so you can invoke it easily. Here are instructions on how to do this:

```
To add a directory like $HOME/bin to the system PATH on Windows,
follow these steps:

    Determine the full path:
    In Windows, $HOME typically refers to your user profile directory.
    For example, if your username is "JohnDoe", the full path might
be:
    C:\Users\JohnDoe\bin
    Open Environment Variables:
        Press Windows key + R
        Type "sysdm.cpl" and press Enter
        Go to the "Advanced" tab
        Click "Environment Variables"
    Edit the PATH variable:
        Under "System variables", find and select "Path"
        Click "Edit"
        Click "New"
        Enter the full path to your bin directory (e.g., C:\Users\
JohnDoe\bin)
```

```
        Click "OK" to close each window
    Apply changes:
        Click "OK" to close all dialog boxes
        Restart any open command prompts or applications for the
changes to take effect
```

Further, in Windows you can associate the .py file in your bin directory so you can simply double-click on it to execute (without any command-line arguments):

To associate .py files with Python, in Windows Explorer:

```
    Right-click on any .py file
    Select "Open with" > "Choose another app"
    Select "Python" from the list (or browse to python.exe if it's not
listed)
    Check the box "Always use this app to open .py files"
    Click "OK"
```

Final Adjustment

If you're using Linux or Mac, set the permissions flags to make the `photorename.py` executable by going to the bin directory—`cd bin`—and typing:

```
chmod 750 photorename.py
```

Specifying Your Target Directory

This script is hardwired to your photo working directory, which it needs to know, including the full path to where it is located.

This can be as simple as making a subdirectory called `working` or `tmp` under MyPhotos or wherever you store you photos on your system. I use the cloud storage service Dropbox for mine, since I work across all three computing environments. I have `Dropbox/Images/tmp` as my target directory under Linux, Mac, and Windows.

To set your own working directory you will need to modify at least one of the following lines in the `photorename.py` file, using Notepad in Windows, TextEdit in Mac, or your choice of plain text editor.

If you use all three operating systems your entry should look somewhat like mine. If you're a Windows user, carefully modify the line that begins `win_targetdir`. For Mac, modify line that begins `mac_targetdir`. For Linux

modify line that begins `linux_targetdir`. The script will determine which operating system you're using and choose the appropriate line found under `##### TARGET DIRECTORIES #####`:

```
##### TARGET DIRECTORIES #####
# Set path to target directory containing files to be renamed
linux_targetdir = "/home/gene/Dropbox/Images/tmp/"
mac_targetdir = "/Users/gene/Dropbox/Images/tmp/"
win_targetdir = "C:/Users/gene/Dropbox/Images/tmp/"
```

As you can see, these are my filepaths to my personal working directory. The pathname is different for each operating system. Replace mine with the one (or more) of yours.

Finally, find the line near the top of the files that reads `photog = "g"` and substitute your initial or initials within the quotes.

Set your own target directory and save the `photorename.py` script.

Using the Script

You're now set to go. Place a bunch of your photo files in your working directory (use copies of these files rather than the originals until you have confidence that this script doesn't do anything untoward to them). Open a terminal on your computer (or 'cmd' in Windows) and type `photorename.py` or `photorename.py $1 $2` substituting photographer's initial(s) for '$1' and your tags for '$2'.

The next step is usually to touch up each photo file with additional or unique tags, using an image viewer.

Last, once your files are fully reviewed and renamed, move the files from your working directory and place them into an appropriate permanent directory. I save my image files by year, so I have directories like 'Dropbox/Images/2003', 'Dropbox/Images/2004/, etc.

Another last thing: if this script hits an image file that doesn't contain exif data, it puts today's date on it instead of no date. I find this handy for images I want to save, such as AI-generated images I've been experimenting with. In general, they don't contain any EXIF data so today's date is close enough for my purposes. Plus appropriate tags that describe the AI image.

About the Programmer

I'm Gene Wilburn, a Canadian writer, photographer, retired IT professional, and occasional folk singer. I have been photographing since age 12 so I curate a lot of scanned and out-of-camera digital images. I have found that the system I advocate here not only works, but works very well for organizing and finding images.

Let me add that I'm not a programming maven. I'm a duffer who writes scripts and small utilities when I can't find ones that already do a particular task I need done, or in this case, a utility that is cross-platform. I hope you enjoy using `photorename.py`. Kudos to you if you want to modify it for your own needs. It's openly open source, with no copyright attached.