File Renaming Process

The program looks for the folder name, as commanded on line 26 [formata]. If found, it attempts to compare the dates on the image files with those created by the program (specified by the user). When executed, the program creates a csv and html file with the user-defined expected date range for each image file and cropped sections of the image files containing the handwritten dates. The goal is then to get the dates on the html file to match the dates written on the original logbook pages, by adding blank files and deleting extra files from the folder. If done correctly, all of the dates in the logbook (program created dates) should match those on the image files (cropped images).

When the dates mismatch, any extra files can go into a discard folder. For our purposes, we’ve just been calling it “ItemXX\_Discards”. For example, if there are blank pages at the start of the html file, those can go into the discard folder. “Extra” files for a given day tend to occur when there are issues with the microfilm page, or when extraneous material (e.g. newspaper clippings) were inserted into the logbook: the microfilm process took several images of each page with the clipping covering portions of the logbook’s pages. The file to be discarded in such a case should be the file with the least amount of information covered by the clipping, usually the image where the clipping is on the right hand side of the ledger. These images should be re-photographed at a later date in collaboration with McGill Archives (Gordon Burr and Lori Podolsky).

If any files are missing, a blank sheet called “missing.jpg” can be inserted in a way where it will go where needed. For example, if a file is missing for the first page of February 1st, 2nd and 3rd and the next file is 200, a “missing.jpg” file can be added that’s called Item\_199\_a to place it before the missing information.

If data is in a 3-day format, use the program called “check\_name\_formata.py”. If the data is in a weekly format, use the program called “check\_name\_formatb.py”. The original program check\_name.py” should be adjusted for each ledger type, especially for the number of days recorded on each page and the number of pages for each date range (e.g. format “A” has two pages of data for each 3-day period). If there are other problems, use your best judgment.

1. Create folder titled “McGill Observatory”, with a folder inside called “1491”
   1. Place the program that will be run inside of this folder. This should be called “check\_name\_formata.py”
2. This guide assumes that python 2.7.6 is being used on UNIX. If you have python 3.4.3, I know that you can still use an older version by simply typing python and not python3. If using windows, you can still follow the links in the instructions
   1. Ensure that pip is installed to your terminal. More information can be found here: <https://pypi.python.org/pypi/pip> and here: <http://stackoverflow.com/questions/9797277/how-to-install-pip-in-a-new-python-installation>
   2. Install PIL or pillow, a module that allows image storage. More information can be found here: <http://pillow.readthedocs.org/en/latest/installation.html> and here: <https://pypi.python.org/pypi/Pillow/2.9.0>
3. Download the folder you want to work with from one drive. I will use Item 52 as an example.
   1. Make sure the folder has a good title. In this case, “Item52\_Reel9\_1900\_Jan-Jun”
   2. Now the folder “Item52…” should be inside “1491” which is inside “McGill Observatory”
4. Create a discard folder for the extra images that will be created. Call this “Item52\_Discards”
5. Before running the program, modify the lines with “image folder =” (~ line 26) and start = datetime.date (~ line 30). In the first line, replace the character text stating “Item…” with the folder name you are working with. In the other line, state the dates that you are working with. In this case I want the first modified line to say “imagefolder = 'Item52\_Reel9\_1990\_Jan-Jun'” and the second modified line to say start = datetime.date(1990, 1, 1)
6. Run the program. A crop folder should be created. In this case, it’s titled “Item52\_Reel9\_1990\_Jan-Jun\_crops”. A csv file “Item20\_Reel4\_1884\_Jan-Ju\_check\_name\_results.csv” and html file “Item20\_Reel4\_1884\_Jan-Ju\_results.html” should also be created
7. Check the HTML file. See if the dates match. Place any extra days in the discard folder and add any missing days with the “missing.jpg” file. Keep rerunning the program until all of the cropped image dates match with the program defined dates.
8. Some of the microfilm reels were photographed “backwards”, with the first image files corresponding to the end of the period (e.g. June 30 or Dec 31) and the last files corresponding to the beginning of the period (Jan 1 or July 1). In this case, “daysperpage” should be set to a negative value (e.g. -3 if there are three days per page).

Crop dimensions for item ranges (to check)

Items 1-5: cropdimensions = (1700, 180, 2550, 360)

Items 6-7: cropdimensions = (1700, 220, 2900, 500)

Items 8-12: cropdimensions =(2500, 200, 4000, 450)

Items 13- : cropdimensions =(1300, 100, 2200, 250)

: