UserGuide

Jump to bottom

davide-romanini edited this page on 4 Feb 2015 · 3 revisions

General Features

Filename Parsing

When an archive is opened, and it has no tags, ComicTagger will attempt to parse the filename to glean some information. Generally, it expects a format similar to this:

```
SERIES_NAME vVOLNUM ISSUENUM (of COUNT) (PUBYEAR).ext
```

or

```
SERIES_NAME (SERIESYEAR) ISSUENUM (of COUNT) (PUBYEAR).ext
```

i.e.

```
Plastic Man v1 002 (1942).cbz
Blue Beetle 02.cbr
Monster Island vol. 2 #2.cbz
Crazy Weird Comics 2 (of 2) (1969).rar
Super Strange Yarns (1957) #92 (1969).cbz
Action Spy Tales v1965 #3.cbr
```

(In this scheme, the PUBYEAR refers to publication year of the issue, and SERIESYEAR as the start year of the volume/series)

Some other notes:

- ComicTagger is happiest when the issue number has a "#" in front of it:

```
Foobar-Man #121 (1974).cbz
```

- The volume name should come after the series name and before the issue number.

```
Foobar-Man Annual v2 #121 (1984).cbz
Foobar-Man Annual Vol. 2 121 (1984).cbz
```

- The first thing the filename parser tries to find is the issue number. If the '#' isn't used, the parser will try to find the most likely number. Generally, any parts of the filename that don't have parentheses around them are considered. The first example is will parse okay:

```
Foobar-Man Annual v2 121 (The Wrath of Foobar-Man, Part 1 of 2).cbz
```

> This second example will not parse OK as the parser doesn't know which number is the issue number:

```
Foobar-Man Annual 121 - The Wrath of Foobar-Man, Part 1 of 2.cbz
```

> You can also use a double dash ("--") or double underscore (" __ ") to precede the part of the filename that won't be parsed. The following example will parse OK, since the text after the double dash will be ignored:

```
Foobar-Man Annual v2 121 -- The Wrath of Foobar-Man, Part 1 of 2.cbz
```

- As mentioned above, most text in parentheses will be ignored. This is also true for text after "--" or " __ ":

```
Monster Island v1 1 (orginal cover) (c2c) .cbz
Monster_Island_v1_2__repaired__c2c.cbz
Monster Island v1 3 (1957) -- The Revenge Of King Klong (noads).cbz
```

> Any text that is not explicitly parsed as series name, volume, issue number, or publication year, or issue count will be saved in the "scan info" field, if enabled.

- There should only be either a volume number or series year in the file name, since these are interpreted by the parser as the same thing. Whether to use start year or sequential volume numbers seems a matter of personal preference, but as even the publishers' use of sequential volume numbers is inconsistent, the start year is more unambiguous.

> The following examples show variations on use of the series year. Any text in the parentheses after the "-" will be ignored by the parse:

```
Dusk Vampire (2010) #1.cbz
Dusk Vampire (2010-) #1.cbz
Dusk Vampire (2010-2012) #1.cbz
```

The cleaner the original filename is, the better ComicTagger is at matching online.

Comic Vine

The online database at Comic Vine (http://comicvine.com) provides a public interface that ComicTagger uses to acquire most of the tagging info. Each user of ComicTagger is recommended to create an account at Comic Vine, and get an "API Key" of their own (at http://comicvine.com/api). The personal key can be added to ComicTagger settings through either the settings dialog, or via the command-line interface.

Though not strictly necessary, if a personal key is not configured, ComicTagger may often fail by using the default application key. This is because Comic Vine implements a per-key limit on API access rates. Since the default application key is shared, it may already be exhausted by other users. At the time of writing, the rate limit is said to be 200 accesses per 15 minutes. Fetching the data to tag a comic can typically involve 2-6 accesses (but can be higher).

Automatic Identification and Tagging

ComicTagger uses an image processing technique to compare the cover image of an archive with images found online to determine the correct metadata.

When an online search and auto-identification is attempted, ComicTagger will use the series name, issue number, and optionally the publication year to find the best match. Once the list of possibilities is narrowed, it will compare cover images to find the right choice.

(If the filename parsing failed to properly fill in the series name and number, you can manually enter those fields before you do the auto-identify.)

Obstacles to this process are alternate covers, duplicate covers (such as multiple editions with the same numbering and virtually identical covers), improperly or poorly named files, and mis-sequenced cover images in the archive.

The settings allow the user to mitigate some of these problems by blacklisting certain publishers, and narrowing the name matching.

Caching

ComicTagger caches online search results and images from ComicVine in a temporary database to improve speed when doing similar searches repeatedly.

RAR Tools

In order to read and write RAR and CBR archives, you will need to download tools from WinRAR. You may need to specify the tool locations via the settings/preferences dialog.

ComicTagger uses the command-line tools "rar" and "unrar", which, despite the warnings of the WinRAR GUI, don't seem to expire. So, no need to pay!

Rename

Renaming based on tag data is available, and a template can be set via the GUI settings dialog, or directly editing the settings file.

The template string accepts the following variables:

```
%series%
%issue%
%volume%
%issuecount%
%year%
%month%
%month_name%
%publisher%
%title%
%genre%
%language_code%
%criticalrating%
%alternateseries%
%alternatenumber%
%alternatecount%
%imprint%
%format%
%maturityrating%
%storyarc%
%seriesgroup%
%scaninfo%
```

For example, the template:

```
%series% #%issue% (%year%)
```

might yield the name:

```
Plastic Man #002 (1944)
```

A "smart text cleanup" can optionally remove extra characters if some of the variables have empty values. For example, if there is no publication year available, the smart cleanup will remove the empty parentheses in the above example.

CBL Transforms

The ComicBookLover tag format (ComicBookInfo or CBI) has less dedicated fields than Comic Vine or the ComicRack format, but it does allow for a generic tag list that can be of any length. In addition, CBI has a unique way of managing credit information that has a concept of "Primary Writer" and "Primary Artist", which can often be guessed from the data at hand.

To try to cope with these idiosyncrasies, ComicTagger has a set of one-way optional "transforms" can be applied to when copying from other tag formats, when fetching data from Comic Vine, or manually via the GUI.

The transforms include:

- Assume the only writer or artist credit is the primary one
- Copying the list of characters to the generic tags
- Copying the list of teams to the generic tags
- Copying the list of locations to the generic tags
- Copying the weblink to the text comments

Settings

ComicTagger keeps the program settings (and caches) in a special folder. On Windows, you might find it in a folder like:

```
C:\Users\usrname\AppData\roaming\ComicTagger
```

This might be a little different, depending on your version of Windows

On Unix systems, (including Mac) something like this:

```
/Users/username/.ComicTagger
```

or

```
/home/username/.ComicTagger
```

The settings for the app affect both GUI and CLI mode. If you're using the GUI, everything in the settings folder can be changed via the app. If only using CLI mode, you might want to edit things such as RAR program location, and identifier settings.

The setting file itself is a standard "ini" format, and very human-readable.

```
[settings]
rar_exe_path = /usr/bin/rar
unrar_exe_path = /usr/bin/unrar

[auto]
last_selected_load_data_style = 0
last_selected_save_data_style = 0
last_opened_folder =
last_main_window_width = 1096
last_main_window_height = 611
last_main_window_x = 252
last_main_window_y = 294
last_form_side_width = 763
last_list_side_width = 309

[identifier]
id_length_delta_thresh = 5
id_publisher_blacklist = Panini Comics, Abril, Scholastic Book Services

[dialogflags]
ask_about_cbi_in_rar = True
show_disclaimer = True

[comicvine]
use_series_start_as_volume = False

[cbl_transform]
assume_lone_credit_is_primary = False
copy_characters_to_tags = False
```

```
copy_teams_to_tags = False
copy_locations_to_tags = False
copy_notes_to_comments = False
copy_weblink_to_comments = False
apply_cbl_transform_on_cv_import = False
apply_cbl_transform_on_bulk_operation = False

[rename]
rename_template = %series% #%issue% (%year%)
rename_issue_number_padding = 3
rename_use_smart_string_cleanup = True
rename_extension_based_on_archive = True
```

GUI User Guide

The GUI is for the most part pretty intuitive. You open files or folders via the menu, or by drag-and-drop, edit the tag fields, and save. But a some portions deserve a few words.

Archive List

The right side of the main window is a list of all the comic files that have been opened by ComicTagger. As each file is read in, its existing metadata and other information are parsed and cached for quick browsing. Changing main selections in the list will cause the form to update, depending on which "Read Style" is selected. A right-click context menu is available for easy access to batch functions.

Multiple archives can be selected for batch operations.

Tag Style Drop-downs

The "Read Style" and "Modify Style" drop-down selections in the upper corner select which tags are shown in the form, and how the tags will be saved.

When the "Modify Style" is changed, certain entry fields will change color, and go inactive if those fields won't be saved for the selected style. For example, the ComicRack tag style doesn't support a field for "Country", so that entry control will be marked red, and made inactive. This block also affects the tag removal menu option, auto-tag, save, and the copy destination tag block.

The "Read Style" affects which tag block shown in the form when the user changes the list selection. This drop-down also affects which tags will be used in renaming, and the source tag block for a copy operation.

Search vs. Auto-Identify vs. Auto-Tag

The ComicTagger GUI offers several different ways to acquire tags online:

The "Search" function uses only the text that has been entered in the "Series" field of the form. It offers an entirely interactive method for tagging a single file. It affects only the data in the form.

"Auto-Identify" uses the "Series", "Issue", and "Year" fields in the form. It will attempt to compare covers of comics online to determine the best match for a single file. Like the search function, it affects only the data in the form.

"Auto-Tag" does not use the form data at all. It is meant to be used with many files at once, and will use any metadata in the file and the file names as search keys. As it matches comics, it will write out the tag automatically.

Page Browser

Selecting the Page Browser menu option/toolbar button will bring up a very simple page broswer. It's meant to allow the user to to inspect pages of the archive to glean hints about metadata, and is in no way meant to be a comic reader.

In addition, double-clicking on cover or page images throughout the app will often bring up a full page view of the image.

Page List Editor

The page list editor tab allows the user to change order that pages are display, and assign different page types. The page list order can be changed by using the up and down buttons, or by dragging and dropping.

CLI User Guide

ComicTagger supports a powerful command-line interface that can be used for batch processing of archives.

On the command-line just invoke the main executable with options.

On Windows the executable path should be something like this:

```
C:\Program Files\ComicTagger\comictagger.exe
```

On Mac something like this:

```
/Applications/ComicTagger.app/Contents/MacOS/ComicTagger
```

> Since that is a little ungainly, you might want to consider running the following so you can just use "ComicTagger" from your shell prompt:

```
sudo mkdir -p /usr/local/bin
sudo ln -s /Applications/ComicTagger.app/Contents/MacOS/ComicTagger  /usr/local/bin
```

If running from python source, just run "comictagger.py"

Here is the help message:

```
Usage: comictagger.py [OPTION]... [FILE LIST]

A utility for reading and writing metadata to comic archives.
```

```
If no options are given, comictagger.py will run in windowed mode

-p, --print                Print out tag info from file.  Specify type
(via -t) to get only info of that tag type
--raw                  With -p, will print out the raw tag block(s)
from the file
-d, --delete               Deletes the tag block of specified type (via -t)
-c, --copy=SOURCE          Copy the specified source tag block to destination style
specified via via -t (potentially lossy operation)
-s, --save                 Save out tags as specified type (via -t)
Must specify also at least -o, -p, or -m
--nooverwrite          Don't modify tag block if it already exists ( relevent for -s or -c )
-1, --assume-issue-one     Assume issue number is 1 if not found ( relevent for -s )
-n, --dryrun               Don't actually modify file (only relevent for -d, -s, or -r)
-t, --type=TYPE            Specify TYPE as either "CR", "CBL", or "COMET" (as either
ComicRack, ComicBookLover, or CoMet style tags, respectivly)
-f, --parsefilename        Parse the filename to get some info, specifically
series name, issue number, volume, and publication
year
-i, --interactive          Interactively query the user when there are multiple matches for
an online search
--nosummary            Suppress the default summary after a save operation
-o, --online               Search online and attempt to identify file using
existing metadata and images in archive. May be used
in conjuntion with -f and -m
--id=ID                Use the issue ID when searching online.  Overrides all other metadata
-m, --metadata=LIST        Explicity define, as a list, some tags to be used
e.g. "series=Plastic Man , publisher=Quality Comics"
"series=Kickers^, Inc., issue=1, year=1986"
Name-Value pairs are comma separated.  Use a "^" to
escape an "=" or a ",", as shown in the example above
Some names that can be used:
series, issue, issueCount, year, publisher, title
-r, --rename               Rename the file based on specified tag style.
--noabort              Don't abort save operation when online match is of low confidence
-e, --export-to-zip        Export RAR archive to Zip format
--delete-rar           Delete original RAR archive after successful export to Zip
--abort-on-conflict    Don't export to zip if intended new filename exists (Otherwise, creates
a new unique filename)
-S, --script=FILE          Run an "add-on" python script that uses the comictagger library for
custom
processing.  Script arguments can follow the script name
-R, --recursive            Recursively include files in sub-folders
-v, --verbose              Be noisy when doing what it does
--terse                Don't say much (for print mode)
--version              Display version
-h, --help                 Display this message
```

Here are some sample usages, that assume running from bash shell on a Mac or other UNIX system. If using Mac or Windows binary versions, substitute the correct ComicTagger command, as mentioned above.

Print out the tags for a single file

```
ComicTagger -p Plastic_Man_001_.zip
```

Print out the ComicRack tags for all CBZ files in the folder

```
ComicTagger -p -t cr *.cbz
```

Delete the ComicBookLover tags for all CBR files in the folder

```
ComicTagger -d -t cbl *.cbr
```

Set the ComicRack tags from online for a single file, using the filename to search

```
ComicTagger -s -t cr -f  -o Plastic_Man_001_.cbr
```

Search online for for a single file, using the specified tags to search, be verbose, and save to ComicRack tags

```
ComicTagger -s -t cr -m "series=Plastic Man,issue=1"  -o -v PlasMan001.cbr
```

Set the ComicBookLover Publisher tag for a list of files

```
ComicTagger -s -t cbl -m "publisher=Quality Comics"   Plastic\ Man*.*
```

As above, but in "dry run" mode, where it just displays what would be saved

```
ComicTagger -n -s -t cbl -m "publisher=Quality Comics"   Plastic\ Man*.*
```

Search online for for a list of files, using the filename to search, interactively ask the user at the end (if needed) and save to ComicRack tags

```
ComicTagger -s -t cr -f  -o -i *.cbr
```

As above, but un-sets any year found in the filenames

```
ComicTagger -s -t cr -f  -o -i  -m "year=" *.cbr
```

Rename a file based on ComicRack tags

```
ComicTagger -r -t cr PlasticManOne.cbz
```

Copy the ComicBookLover tags in an archive to ComicRack tags

```
ComicTagger -c cbl -t cr PlasticManOne.cbz
```

Export all CBR files in folder to Zip format

```
ComicTagger -e *.cbr
```

Recursively tag all comics (without ComicRack tags already) under the current folder

```
ComicTagger -R -s -f -o -t cr --nooverwrite -i -v /path/to/comics
```

Run a custom python script using comictaggerlib:

```
ComicTagger -S /path/to/script.py
```

The following examples make use of the "find" command do do recursive searches:

Recursively tag all comics (without ComicRack tags already) under the current folder

```
find . -name \*.cb\* |  sed 's/.*/"&"/' |  xargs ComicTagger -s -f -o -t cr --nooverwrite -i
```

Recursively copy all ComicRack tags to CBL tags

```
find . -name \*.cb\* |  sed 's/.*/"&"/' |  xargs ComicTagger -c cr -t cbl
```

▸ **Pages** 9

#Table of contents

Clone this wiki locally