# Beets

https://github.com/beetbox/beets/

Note: Updating python requests will cause conflicts with the distro installed pip version. Best to uninstall it completely and get the latest using easy_install:

```
#mint21
sudo apt-get install libchromaprint1 mp3val flac python3-acoustid
sudo apt-get remove python-pip
sudo apt-get install python3-setuptools

sudo easy_install pip
pip install beets
pip install requests
pip install beets[fetchart,lyrics,lastgenre]
pip install beets-yearfixer
pip install https://github.com/ocelma/python-itunes/archive/master.zip
pip install discogs-client
pip install beets-web-import
pip install python3-discogs-client
# see notes below to patch copyartifacts
pip install beets-copyartifacts

#Buggy as of 19/Sept/2020:
pip install beets-copyartifacts3
pip install beets-extrafiles

#Mint22:
sudo apt-get install python3-acoustid libchromaprint1 mp3val flac
pipx install beets
pipx inject beets flask discogs_client requests pylast pyacoustid pycairo
pygobject python3-discogs-client discogs-client beets-copyartifacts beets-
extrafiles
pipx inject beets beets-yearfixer
pipx inject beets git+https://github.com/edgars-supe/beets-importreplace.git
#for lyrics plugin:
pipx inject beets langdetect bs4
```

to upgrade beets going forward, use:

```
pip install -U beets
```

As beets 1.6.0 is a bit buggy and 3 years old, use latest source snapshot tarball installation:

```
pip uninstall beets
pip install https://github.com/beetbox/beets/tarball/master
```

Uninstall plugins from pipx:

```
Show all pipx envs and injected packages to find exact package names
pipx list --include-injected

then run:
pipx uninject beets beets-web-import
```

# shell completion

add to .bashrc or similar:

```
eval "$(beet completion)"
```

# copyartifacts python3 patch

```
2to3-2.7 -w ~/.local/lib/python3.10/site-packages/beetsplug/copyartifacts.py

#Mint22:
sudo apt-get install 2to3
2to3 -w ~/.local//share/pipx/venvs/beets/lib/python3.12/site-
packages/beetsplug/copyartifacts.py
```

then adjust as following:

```
#line 104:
                file_ext = os.path.splitext(filename)[1].decode('utf8')
#line 143:
                print('   ', str(os.path.basename(f.decode('utf8'))))
#line 147:
        print('Copying artifact:
{0}'.format(os.path.basename(dest_file.decode('utf8'))))
#line 156:
        print('Moving artifact:
{0}'.format(os.path.basename(dest_file.decode('utf8'))))

#add line 127
                dest_file = beets.util.bytestring_path(dest_file)
```

Change the init:

```
vi ~/.local/lib/python3.10/site-packages/beets/autotag/__init__.py
```

```
#change line 75 from
def unidecode_punc_only(text):
#to
```

```
def unicode_punc_only(text):
```

# Unicode in Tag Patch

NOTE: use import-replace plugin instead!

MusicBrainz uses unicode characters for apostrophes, dashes etc rather than ascii versions. A beets plugin for filenames (asciify) solves some issues with this, but tags are unaffected. This patch for v1.5.0/v1.6.0 on python 3.8 solves this for imports:

```
pip install unidecode
```

```
#beets version 1.6.0
cat > beets-unidecode.patch << EOF
20a21
> from unidecode import unidecode
73a75,85
> def unidecode_punc_only(text):
>     """Change unicode punctuation to ascii equivalent.
>     """
>     result = u""
>     for character in text:
>         if character.isalpha():
>             result += character
>         else:
>             result += unidecode(character)
>     return result
>
77,80c89,92
<     item.artist = track_info.artist
<     item.artist_sort = track_info.artist_sort
<     item.artist_credit = track_info.artist_credit
<     item.title = track_info.title
---
>     item.artist = unicode_punc_only(track_info.artist)
>     item.artist_sort = unicode_punc_only(track_info.artist_sort)
>     item.artist_credit = unicode_punc_only(track_info.artist_credit)
>     item.title = unicode_punc_only(track_info.title)
105,110c117,122
<             item.artist = (track_info.artist_credit or
<                            track_info.artist or
<                            album_info.artist_credit or
<                            album_info.artist)
<             item.albumartist = (album_info.artist_credit or
<                                 album_info.artist)
---
>             item.artist = (unicode_punc_only(track_info.artist_credit) or
>                            unicode_punc_only(track_info.artist) or
```

```
>                             unicode_punc_only(album_info.artist_credit) or
>                             unicode_punc_only(album_info.artist))
>             item.albumartist =
(unicode_punc_only(album_info.artist_credit) or
>                                 unicode_punc_only(album_info.artist))
112,113c124,125
<             item.artist = (track_info.artist or album_info.artist)
<             item.albumartist = album_info.artist
---
>             item.artist = (unicode_punc_only(track_info.artist) or
unicode_punc_only(album_info.artist))
>             item.albumartist = unicode_punc_only(album_info.artist)
116c128
<         item.album = album_info.album
---
>         item.album = unicode_punc_only(album_info.album)
119,123c131,135
<         item.artist_sort = track_info.artist_sort or
album_info.artist_sort
<         item.artist_credit = (track_info.artist_credit or
<                               album_info.artist_credit)
<         item.albumartist_sort = album_info.artist_sort
<         item.albumartist_credit = album_info.artist_credit
---
>         item.artist_sort = unicode_punc_only(track_info.artist_sort) or
unicode_punc_only(album_info.artist_sort)
>         item.artist_credit = (unicode_punc_only(track_info.artist_credit)
or
>                               unicode_punc_only(album_info.artist_credit))
>         item.albumartist_sort = unicode_punc_only(album_info.artist_sort)
>         item.albumartist_credit =
unicode_punc_only(album_info.artist_credit)
149c161
<         item.title = track_info.title
---
>         item.title = unicode_punc_only(track_info.title)
EOF

sudo patch -b ~/.local/lib/python3.10/site-
packages/beets/autotag/__init__.py beets-unidecode.patch
```

# Show path in duplicate import warning

```
vi ~/.local/lib/python3.10/site-packages/beets/ui/commands.py
#add as line 476 in beets 1.6.0, around line 740 in beets 1.6.1:
        summary_parts.append(displayable_path(item.path))
```

# Beets tips

http://ngokevin.com/blog/beets/
https://beets.readthedocs.io/en/v1.4.9/guides/advanced.html#fetch-album-art-genres-and-lyrics
https://beets.readthedocs.io/en/v1.4.9/guides/tagger.html
https://beets.readthedocs.io/en/v1.4.9/plugins/chroma.html
https://beets.readthedocs.io/en/v1.4.9/plugins/fromfilename.html

Using beets as an autotagger, without importing the files: i.e. just point beets to a directory, have it tag with data from MusicBrainz, but don't save anything to the database, don't rename or move the files, etc.

There's no option to avoid writing to the database - but it can be configured to not move or copy the files:

```
import:
    copy: no
    move: no
```

Then you can just ignore that the database exists at all. Or just `alias beet='beet ; rm ~/.config/beets/library.db` or something to delete it every time.

If you want a temporary configuration for any reason, you can always use `beet -c /path/to/file.yaml` instead of copying around files.

# Config File

Default config file of beets:
https://raw.githubusercontent.com/beetbox/beets/master/beets/config_default.yaml

Personal adjustment:

~/.config/beets/config.yaml

```
#
# Beets configuration file
#
# Tip: Import works best by first importing albums, ideally one album
at a time, then singles to address duplicates properly
#

library: /home/wuff/.config/beets/beetslibrary.db
directory: /media/music/beets2

import:
    # write metadata to music files
    write: yes
```

```
    # copy imported files from source to the music directory
    copy: yes
    # move imported files from source to the music directory - takes
precedence over copy!
    move: yes
    link: no
    hardlink: no
    #deprecated by move function
    #delete: no
    resume: ask
    incremental: no
    incremental_skip_later: no
    from_scratch: no
    quiet_fallback: skip
    none_rec_action: ask
    timid: no
    #log:
    log: /home/wuff/.config/beets/beetslog.txt
    # use auto-tagging where possible do not require confirmation on
strong matches
    autotag: yes
    # no output, use quiet_fallback - skip - as default action
    quiet: no
    #default to full album import, singleton mode via command line -s
switch
    singletons: no
    # default selected action on command line for return for a match
query
    default_action: apply
    languages: []
    detail: no
    #multi disk albums split in multiple directories can be seen as
flat using cmd option --flat
    flat: no
    group_albums: no
    pretend: false
    search_ids: []
    duplicate_action: ask
    #duplicate verbose uses format_item settings
    duplicate_verbose_prompt: yes
    bell: no
    set_fields: {}
    art: yes


# files matching these patterns are deleted from source after import
clutter: ["Thumbs.DB", ".DS_Store", "desktop.ini", "*.m3u", "*.nfo",
"*.pls", "*.torrent", "cover.jpg"]
# files/directories matching one of these patterns are ignored during
import
ignore: [".*", "*~", "System Volume Information", "lost+found",
```

```
"cover.jpg"]
ignore_hidden: yes
# replace special characters in generated filenames
replace:
    '[\\/]': '-'
    '^\.': _
    '[\x00-\x1f]': _
    '[<>\?\*\|]': _
    '[:]': ' -'
    # dot at end of directories causes problems with some clients on
samba shares
    '\.$': _
    '\s+$': ''
    '^\s+': ''
    '^-': _
    # no special quotes thank you
    '[\u2018\u2019]': ''''
    '[\u201c\u201d]': '"'
    '[\x91\x92]': "'"
    # double quotes to single quote
    '"': ''''

#replacement character for path separator (forward/backslack),
precedence over replace function
path_sep_replace: '-'
#replacement character for drive separator (colon?), NO precedence over
replace function
drive_sep_replace: '-'

#default false, set to yes, all special characters or accents are
converted to ascii equivalent.
#avoids having to wrap all paths with asciify function
asciify_paths: true
art_filename: folder  # results in "folder.jpg", default "cover.jpg"
max_filename_length: 0  # 0 unlimited

#add more details for duplicate album names
aunique:
    keys: albumartist album
    disambiguators: albumtype year label catalognum albumdisambig
releasegroupdisambig
    bracket: '[]'

overwrite_null:
  album: []
  track: []

plugins:
[fetchart,lyrics,lastgenre,embedart,bucket,the,chroma,ftintitle,discogs
,scrub,inline,fromfilename,edit,importadded,duplicates,replaygain,web,b
pd,yearfixer,zero,badfiles,importreplace]
```

```
#copyartifacts broken
pluginpath: []
threaded: yes
timeout: 5.0
# per disc numbering no = 2nd disk starts with N+1 track number
# if set to yes, second disk track starts at 1 again.
# If "per disk", make sure tracknames do not collide ("paths" setting)
and include disks in path (and/or inline plugin).
per_disc_numbering: yes
verbose: 0 #for debug info. equivalent to -v. can be used twice for
even more details -vv
terminal_encoding: utf8
# use the release-date of the original (first) release of an album?
original_date: yes
artist_credit: no
id3v23: no
va_name: "Various Artists"

ui:
    terminal_width: 80
    length_diff_thresh: 10.0 #undocumented
    color: yes
    colors:
        text_success: green
        text_warning: yellow
        text_error: red
        text_highlight: red
        text_highlight_minor: lightgray
        action_default: turquoise
        action: blue

#Define how to show items using the list command (formerly
list_format_item/list_format_album)
format_item: %upper{$artist} - $album - $track. $title ($length)
format_album: %upper{$albumartist} - $album [$year] ($totaltracks)
#format_item: $artist - $album - $title
time_format: '%Y-%m-%d %H:%M:%S'

#Track length is displayed as "M:SS" rather than a raw number of
seconds. Queries on track length also accept this format. To use raw
seconds, set format_raw_length to yes
format_raw_length: no

#Default sort order when fetching items from database
sort_album: albumartist+ album+
sort_item: artist+ album+ disc+ track+
sort_case_insensitive: yes

#used by inline plugin to define specific template variables for paths
item_fields:
    disc_and_track: f'{disc:02}-{track:02}' if disctotal > 1 else
```

```
f'{track:02}'
    artist_and_title: f'{artist} - {title}' if artist != albumartist
else title
    original_date: "'-'.join(filter(None, (original_year and
f'{original_year:04}', original_month and f'{original_month:02}',
original_day and f'{original_day:02}')))"
    album_year: f'{year:04}'
album_fields:
    totaltracks: str(items[0].tracktotal) + ' tracks'
    album_ends_with_ep: 1 if str({album})[-2:] == "EP" else 0

# Paths and filenames for music files relative to music directory
# setting asciify to true in general settings avoids having to wrap
individual paths with %asciify{} function.
# In addition to default, comp, and singleton, you can condition path
queries based on beets queries (eg albumtype:soundtrack). The queries
are tested in the order they appear in the configuration file.
paths:
    default:
Album/%bucket{%the{$albumartist},alpha}/%the{$albumartist}/$albumartist
- $album%aunique{} [$year]/$albumartist - $disc_and_track - $title
    singleton: Non-Album/%bucket{%the{$artist}}/$artist - $title
    albumtype:ep:
Album/%bucket{%the{$albumartist},alpha}/%the{$albumartist}/$albumartist
- $album%aunique{}%if{$album_ends_with_ep,, EP} [$year]/$albumartist -
$disc_and_track - $title
    albumtype:soundtrack: Soundtracks/$album%aunique{}
[$year]/$disc_and_track - $artist - $title
    comp:
Compilations/%bucket{$year}/%the{$album%aunique{}}/$disc_and_track -
$artist - $title

statefile: state.pickle

musicbrainz:
    host: musicbrainz.org
    ratelimit: 1
    ratelimit_interval: 1.0
    searchlimit: 20
    extra_tags: []

match:
    strong_rec_thresh: 0.1 # match 90% or better for auto import
(default is 0.04 = 96%)
    medium_rec_thresh: 0.25
    rec_gap_thresh: 0.25
    max_rec:
        missing_tracks: medium
        unmatched_tracks: medium
    #
https://discourse.beets.io/t/can-beets-fix-tracks-which-are-wrongly-nam
```

```
ed/671/4
    distance_weights:
        source: 2.0
        artist: 3.0
        album: 3.0
        media: 1.0
        mediums: 1.0
        year: 1.0
        country: 0.5
        label: 0.5
        catalognum: 0.5
        albumdisambig: 0.5
        album_id: 5.0
        tracks: 4.0 #default 2
        missing_tracks: 0.9
        unmatched_tracks: 0.4 #default 0.6
        track_title: 3.0
        track_artist: 2.0
        track_index: 1.0
        track_length: 2.0
        track_id: 5.0
    preferred:
        countries: ['XW', 'US', 'GB|UK', 'JP', 'DE']
        media: ['Digital Media', 'CD', 'File']
        original_year: yes
    ignored: []
    required: []
    #ignore matches from video sources instead of audio
    ignored_media: ['Data CD', 'DVD', 'DVD-Video', 'Blu-ray', 'HD-DVD',
                    'VCD', 'SVCD', 'UMD', 'VHS']
    ignore_data_tracks: yes
    ignore_video_tracks: yes
    track_length_grace: 10
    track_length_max: 30

embedart:
    auto: yes
    compare_threshold: 20
    ifempty: no  #only embed if empty: default no
    maxwidth: 800
    remove_art_file: no

fetchart:
    auto: yes
    cover_names: front folder cover album
    enforce_ratio: 2%

bucket:
    bucket_alpha: ['0-9', 'A-C', 'D-F', 'G-J', 'K-N', 'O-R', 'S', 'T-Z'
]
    bucket_year:  ['1960s', '1970s', '1980s', '1990s', '2000s',
```

```
'2010s', '2020s']
    bucket_alpha_regex:
        '0-9': ^[^a-zA-Z]

lyrics:
    auto: yes
    on_import: true
    processes: 10
    force: no
    fallback: ''

#moves featured artists from artists to the title field
ftintitle:
    auto: yes
    format: feat. {0}

#Scrub plugin removes all non-beet tags
scrub:
    auto: no

#copies additional files from source folder (except ignored files)
copyartifacts:
    extensions: .cue .jpg .gif .png .pdf
    print_ignored: yes

edit:
    itemfields: track title artist album year albumartist artist_sort
artist_credit
    albumfields: album albumartist year

importadded:
    preserve_mtime: yes #after import reset mtime to original value
    preserve_write_mtime: yes #after writing to files reset mtime

duplicates:
    # creating checksums is expensive
    #checksum: no
    #By default, the tie-breaking procedure favors the most complete
metadata attribute set. If you would like to consider the lower
bitrates as duplicates, for example, set
    tiebreak: { items: [bitrate] }

chroma:
    auto: yes

#Plugin needs gstreamer and python-gi installed, or mp3gain or aacgain
replaygain:
    auto: yes #no=disable during import
    backend: gstreamer
    overwrite: yes #re-analyze files with replaygain tag default no
    targetlevel: 89 #decibel target level default 89
```

```
    #manual analysis  beet replaygain -wf /my/music/directory
    #Note this takes about 10 seconds for 7 files which is about 6h for
15k songs.
web:
    host: 0.0.0.0
    port: 8337

bpd:
    host: 127.0.0.1
    port: 6600
    password: seekrit
    volume: 30

acoustid:
    apikey: 9NQqkUUg6p

# null/remove specific tag fields on import (not when importing as-is)
# regex condition on comments field, this removes useless comments:
zero:
    fields: comments
    comments: [EAC, LAME, from.+collection, 'ripped by']
    update_database: true

# check integrity of files on import, note that mp3val
# required for mp3s: apt-get install mp3val
# mp3val is quite verbose, multiple stream errors indicate issues
badfiles:
    check_on_import: yes
    commands:
      mp3: mp3val -f -si -nb -t

#replace unicode apostrophes from mb with ascii version in tags
importreplace:
  replacements:
    - item_fields: title artist artist_sort artist_credit
      album_fields: album artist artist_sort artist_credit
      replace:
        '[\u2018-\u201B]': ''''
        '[\u201C-\u201F]': '"'
        '[\u2010-\u2015]': '-'
```

# Path template information

https://beets.readthedocs.io/en/v1.4.7/reference/pathformat.html#template-functions

usage:

```
beet import /media/downloads/Thirty\ Seconds\ to\ Mars\ -\ AMERICA\ \(2018\)
```

To rename files/folders after change in path/filename config, use:

```
beet move
```

To use a separate config file use:

```
beet -c beets-wulf-config.yaml import xxx
```

Dry run/Pretend:

```
beet move -p
```

Update db after deleting files

```
beet update
```

Update db after changing ID3 tags (singleton example)

```
beet import -A -C -W -I -s /media/music/beets2/Non-Album/
```

Update id3 tags after changing db

```
beet write
```

# Addons

https://github.com/martinkirch/drop2beets

https://amp.reddit.com/r/airsonic/comments/f1yb86/beets_plugin_for_trigger_library_update/

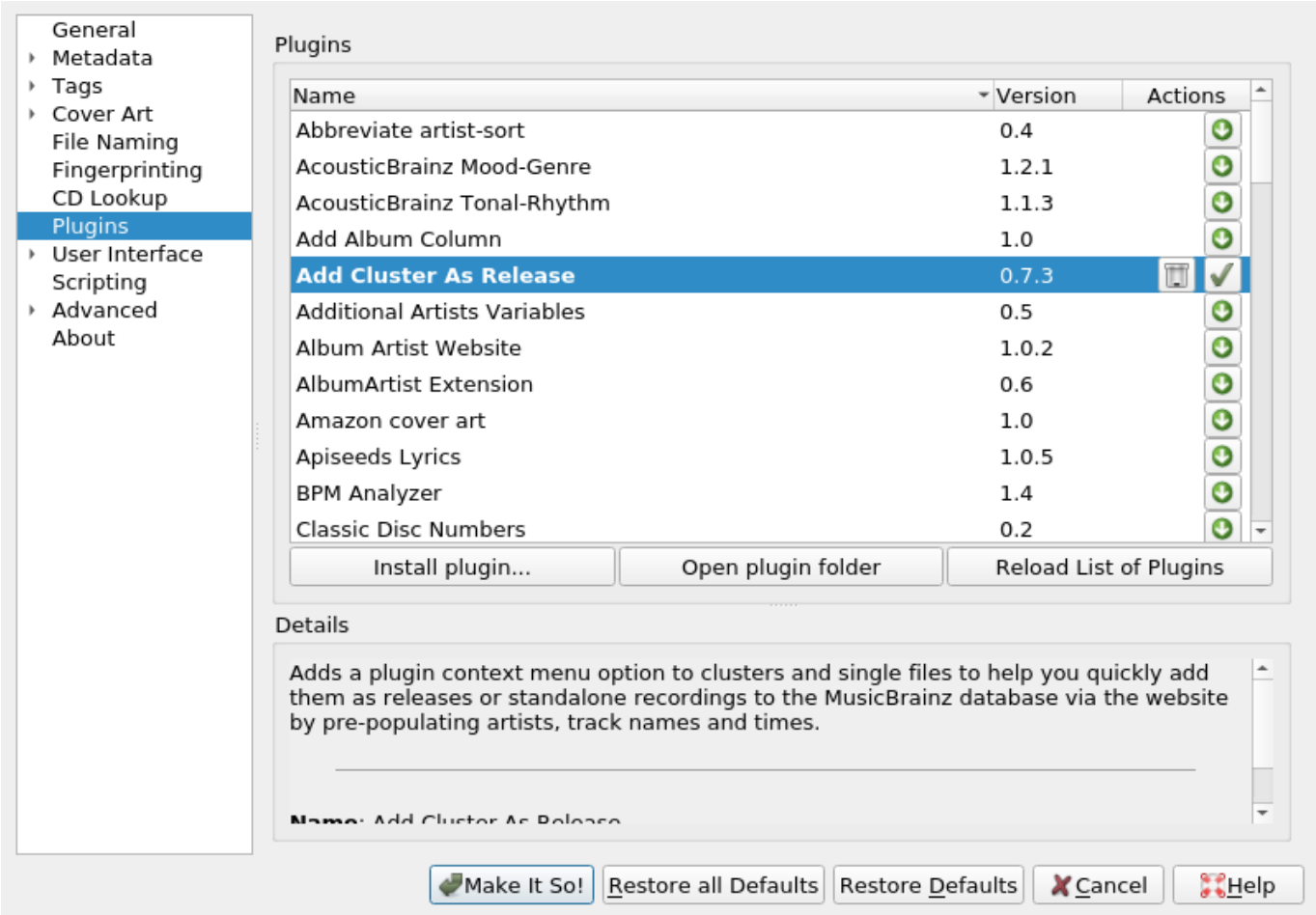
# Submit album to MusicBrainz

https://thesynack.com/posts/managing-music/

Although it is possible to manually add albums (MusicBrainz calls them releases) to MusicBrainz, there is a much easier method: MusicBrainz's GUI music tagger, Picard. This tagger is also quite capable, and some may prefer its GUI interface if the CLI scares you.

Picard offers a fast way to contribute releases to MusicBrainz. To use it, go to the plugins settings inside of Picard and install Add Cluster As Release.

installing the plugin




With the plugin installed:

1. Follow the quick-start instructions to create clusters out of music you wish to contribute.
2. Right-click and select plugins → Add Cluster As Release. This will pull up the contribution form in your browser with tons of information already filled out.
3. Fill in any missing information you can find.
4. Submit the edit.

After a few moments, try re-tagging the album with beets. If everything worked properly, beets should find the release you contributed and finish tagging it. Repeat this process for any albums you wish to add to MusicBrainz.

# beets web

depends on flask

```
pip install flask
```

add web to plugins in beets config.yaml Define port in [web] section

```
[web]
host: 127.0.0.1 #or 0.0.0.0
port: 8337
```

for a remote web interface, CORS is required

```
pip install flask-cors
```

start server on localhost:8337 using

```
beet web
```

or

```
beet web hostname port
```

https://github.com/irskep/summertunes

# beet edit

edit plugin: add config option: editor_command

vi ~/.local/lib/python3.10/site-packages/beetsplug/edit.py: def edit, line 45

```
if self.config['editor_command'].as_str():
  cmd = util.shlex_split(self.config['editor_command'].as_str())
else:
  cmd = util.shlex_split(util.editor_command())
```

Problem: editor env variable not always read?

```
echo $EDITOR
vi

beet edit --all george michael sun go down
No changes; aborting.
```

but:

```
EDITOR=vi beet edit --all george michael sun go down

env | grep EDITOR

export EDITOR=vi

env | grep EDITOR
EDITOR=vi
```

# Album Art Downloader

https://sourceforge.net/projects/album-art/

https://community.metabrainz.org/t/update-cover-art-only/513486/7

Force replace coverart of album:

```
beet fetchart -f carrie underwood carnival ride
beet clearart carrie underwood carnival ride
beet embedart carrie underwood carnival ride
```

# Puddletag

ID3 Tag editor with embed cover support

https://docs.puddletag.net/index.html

```
sudo apt-get install puddletag
```

# KID3 QT

ID3 Tag editor with embed cover support

```
sudo add-apt-repository ppa:ufleisch/kid3
sudo apt-get update
sudo apt-get install kid3-qt
```

# Find/reimport items

To find db entries not linked to musicbrainz, you can use a query.

You can match an empty field with regular expressions:

```
beet ls mb_albumid::^$
```

http://beets.readthedocs.org/en/v1.2.1/reference/query.html#regular-expressions

Additionally, there's no need to move unmatched files to a separate directory to re-import them. You can use the -L flag to re-import albums matching a query:

```
beet imp -L mb_albumid::^$
```

# FtInTitle notes

https://beets.readthedocs.io/en/stable/plugins/ftintitle.html

The FtInTitle plugin identifies the main artist using the albumartist field and compares this with the artist field. If they are identical, the addon will not do anything. The albumartist needs to contain the main artist only.

# yearfixer

https://github.com/adamjakab/BeetsPluginYearFixer https://pypi.org/project/beets-yearfixer/ finds and sets the original year. usage:

```
beet yearfixer [options] [QUERY...]
```

# importreplace

A plugin for beets to perform regex replacements during import, particularly useful to replace unicode apostrophes, double quotes, single quotes and hyphens/en dash from musicbrainz db with ascii equivalents.

https://github.com/edgars-supe/beets-importreplace

Installation:

```
pip install git+https://github.com/edgars-supe/beets-importreplace.git
```

add config and add importreplace to plugins config string:

```
#replace unicode apostrophes from mb with ascii version in tags
importreplace:
  replacements:
    - item_fields: title artist artist_sort artist_credit
      album_fields: album artist artist_sort artist_credit
      replace:
        '[\u2018-\u201B]': '''
        '[\u201C-\u201F]': '"'
        '[\u2010-\u2015]': '-'
```

As this plugin works for import only, use the following to retag/reimport after installation, make sure to copy&paste as the below uses the unicode characters:

```
#for albums:
beet import -L ":[’”-——]"

#for singletons:
beet import -L -s ":[’”-——]"
```

From:
http://wuff.dyndns.org/ - **Wulf's Various Things**

Permanent link:
**http://wuff.dyndns.org/doku.php?id=config:beets&rev=1763899314**

Last update: **2025/11/23 12:01**