

D: 21:13:11,496 tagger.__init__:315: Starting Picard from 'C:\\Program Files\\MusicBrainz Picard\\picard\\tagger.pyc'

D: 21:13:11,496 tagger.__init__:316: Platform: Windows-10-10.0.26200-SP0
CPython 3.8.10

D: 21:13:11,497 tagger.__init__:318: Versions: Picard 2.13.3, Python 3.8.10, PyQt 5.15.11, Qt 5.15.2, Mutagen 1.47.0, Discid discid 1.2.0, libdiscid 0.6.4, astrcmp Python, SSL OpenSSL 1.1.1b 26 Feb 2019

D: 21:13:11,498 tagger.__init__:319: Configuration file path:
'C:/Users/smithar/AppData/Roaming/MusicBrainz/Picard.ini'

D: 21:13:11,498 tagger.__init__:321: User directory:
'C:\\Users\\smithar\\AppData\\Local\\MusicBrainz\\Picard'

D: 21:13:11,498 tagger.__init__:322: System long path support: False

D: 21:13:11,498 tagger.__init__:325: Qt Env.: QT_PLUGIN_PATH='C:\\Program Files\\MusicBrainz Picard\\PyQt5\\Qt5\\plugins'

D: 21:13:11,499 i18n.setup_gettext:153: UI language: system

D: 21:13:11,499 i18n._log_lang_env_vars:138: Env vars:

D: 21:13:11,499 i18n.setup_gettext:161: Trying locales: ['en_US']

D: 21:13:11,539 i18n.setup_gettext:167: Set locale to: 'en_US'

D: 21:13:11,539 i18n.setup_gettext:178: Using locale: 'en_US'

D: 21:13:11,558 i18n._load_translation:125: Loading gettext translation for picard, localedir='C:\\Program Files\\MusicBrainz Picard\\locale', language='en_US'

D: 21:13:11,558 i18n._load_translation:128: [Errno 2] No translation file found for domain: 'picard'

D: 21:13:11,559 i18n._load_translation:125: Loading gettext translation for picard-attributes, localedir='C:\\Program Files\\MusicBrainz Picard\\locale', language='en_US'

D: 21:13:11,559 i18n._load_translation:128: [Errno 2] No translation file found for domain: 'picard-attributes'

D: 21:13:11,560 i18n._load_translation:125: Loading gettext translation for picard-constants, localedir='C:\\Program Files\\MusicBrainz Picard\\locale', language='en_US'

D: 21:13:11,560 i18n._load_translation:128: [Errno 2] No translation file found for domain: 'picard-constants'

D: 21:13:11,561 i18n._load_translation:125: Loading gettext translation for picard-countries, localedir='C:\\Program Files\\MusicBrainz Picard\\locale', language='en_US'

D: 21:13:11,561 i18n._load_translation:128: [Errno 2] No translation file found for domain: 'picard-countries'

D: 21:13:11,562 i18n.setup_gettext:201: _ = <bound method NullTranslations.gettext of <gettext.NullTranslations object at 0x00000161E4614BB0>>

D: 21:13:11,562 i18n.setup_gettext:202: N_ = <function <lambda> at 0x00000161E4E11280>

D: 21:13:11,563 i18n.setup_gettext:203: ngettext = <bound method NullTranslations.ngettext of <gettext.NullTranslations object at 0x00000161E4614BB0>>

D: 21:13:11,563 i18n.setup_gettext:204: gettext_countries = <bound method NullTranslations.gettext of <gettext.NullTranslations object at 0x00000161E6570D30>>

D: 21:13:11,564 i18n.setup_gettext:205: gettext_attributes = <bound method NullTranslations.gettext of <gettext.NullTranslations object at 0x00000161E6570C70>>

D: 21:13:11,564 i18n.setup_gettext:206: pgettext_attributes = <bound method NullTranslations.pgettext of <gettext.NullTranslations object at 0x00000161E6570C70>>

D: 21:13:11,609 webservice._network_accessible_changed:388: Network accessible requested: 1, actual: 1

D: 21:13:11,620 webservice.set_cache:410: NetworkDiskCache dir: 'C:/Users/smithar/AppData/Local/MusicBrainz/Picard/cache/network/' current size: 67.5 kB max size: 100 MB

D: 21:13:11,621 pluginmanager.load_plugins_from_directory:264: Looking for plugins in directory 'C:\\Users\\smithar\\AppData\\Local\\MusicBrainz\\Picard\\plugins', 0 names found

I: 21:13:11,622 pluginmanager.load_plugins_from_directory:252: Plugin directory 'plugins' doesn't exist

D: 21:13:11,630 ui/playertoolbar.__init__:91: Internal player: QtMultimedia available, initializing QMediaPlayer

D: 21:13:11,701 ui/playertoolbar.__init__:98: Internal player: available, QMediaPlayer set up

D: 21:13:13,162 tagger.main:1596: Looking for Qt locale en_US in C:/Program Files/MusicBrainz Picard/PyQt5/Qt5/translations

I: 21:13:13,208 browser/browser.start:121: Starting the browser integration (127.0.0.1:8000)

D: 21:13:13,244 ui/mainwindow.auto_update_check:1785: Skipping startup check for program updates. Today: 2026-01-27, Last check: 2026-01-27 (Check interval: 7 days), Update level: 0 (stable)

D: 21:13:13,249 config.event:261: Config file update requested on thread 22376

D: 21:13:13,285 webservice/ratecontrol.get_delay_to_next_request:122: ('picard.musicbrainz.org', 443): First request

D: 21:13:13,286 webservice/ratecontrol.increment_requests:147: ('picard.musicbrainz.org', 443): Incrementing requests to: 1

D: 21:13:14,854 webservice/ratecontrol.decrement_requests:155: ('picard.musicbrainz.org', 443): Decrementing requests to: 0

D: 21:13:14,855 webservice._handle_reply:559: Received reply for https://picard.musicbrainz.org/api/v2/plugins/ -> HTTP 200 (OK)

D: 21:13:14,857 webservice._handle_reply:572: Response received: {'plugins': {'abbreviate_artistsort': {'api_versions': ['1.0', '2.0'], 'author': 'Sophist', 'description': '<p>Abbreviate Artist-Sort and Album-Artist-Sort Tags.\ne.g. "Vivaldi, Antonio" becomes "Vivaldi, A."\nThis is particularly useful for classical albums that can have a long list of artists.\n%artistsort% is abbreviated into %_artistsort_abbrev% and\n%albumartistsort% is abbreviated into %_albumartistsort_abbrev%.</p>', 'files': {'abbreviate_artistsort.py': 'e076fd7cfbc3b0ced5d2d08ab89749a3'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Abbreviate artist-sort', 'version': '0.4.1'}, 'acousticbrainz': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.5', '2.6', '2.7'], 'author': 'Wargreen <wargreen@lebib.org>, Hugo Geoffroy "pistache" <pistache@lebib.org>, Philipp Wolfer <ph.wolfer@gmail.com>, Regorxxx <regorxxx@protonmail.com>', 'description': '<p>Tag files with tags from the AcousticBrainz database, all highlevel classifiers\nand tonal/rhythm data.\n

\nBy default, only simple mood and genre information is saved, but the plugin can\nbe configured to include all highlevel data.\n

\nBased on code from Andrew Cook, Sambhav Kothari\n

\nWARNING: Experimental plugin. All guarantees voided by use.</p>', 'files': {'__init__.py': 'c731ef9137dfacea18950c9f95bf9453', 'ui_options_acousticbrainz_tags.py': '651c243eeb07dfd1f19e1cfdac9569c6', 'ui_options_acousticbrainz_tags.ui': '571879d618cd4aafb91b717126e7d2ae'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'AcousticBrainz Tags', 'version': '2.2.3'}, 'acousticbrainz_tonal-rhythm': {'api_versions': ['2.0'], 'author': 'Sophist, Sambhav Kothari', 'description': '<p>Add's the following tags:</p>\n\nKey (in ID3v2.3 format)\nBeats Per Minute (BPM)\n\n<p>from the AcousticBrainz database.

\nThis plugin is deprecated, please consider using the AcousticBrainz Tags\nplugin instead.</p>', 'files': {'acousticbrainz_tonal-rhythm.py': '1073fa351166cf596726ae38e7de35eb'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'AcousticBrainz Tonal-Rhythm', 'version': '1.1.6'}, 'add_to_collection': {'api_versions': ['2.0'], 'author': 'Dvir Yitzchaki (dvirtz@gmail.com)', 'description': '<p>Adds any saved release to one of your user collections</p>', 'files': {'README.md':

```

'2d5ebcb30d54a084f3fe90bdac5a50b8', '__init__.py':
'a95aec7f9f14f58fdd539d36db630d9', 'assets/settings.png':
'ea39fbf33350397f0a42599264fb6086', 'manifest.py':
'225ec75ba1b3de0c076af4e718559664', 'options.py':
'127f03004a9bd62871f344bfbab1c93a', 'override_module.py':
'0226c44571c6f29f0e5f26ba5a10f176', 'post_save_processor.py':
'34ad35e4321737dcff3dc65bb081e267', 'settings.py':
'40cfd22d3ad5df1ea07f02a02058b090', 'ui_add_to_collection_options.py':
'c2bfd5a180914a38c4f916727316e5f5'}, 'license': ['MIT'], 'license_url':
'https://spdx.org/licenses/MIT.html', 'name': 'Add to Collection', 'version': '0.1.2'},
'additional_artists_details': {'api_versions': ['2.0', '2.1', '2.2', '2.7', '2.8', '2.11'], 'author':
'Bob Swift (rdsdswift)', 'description': '<p>This plugin provides specialized album and
track variables with artist details for use in tagging and naming scripts. Note that this
creates\nadditional calls to the MusicBrainz API for the artist and area information,
and this will slow down processing. This will be particularly\nnoticable when there
are many different album or track artists, such as on a [Various Artists] release.
There is an option to disable track\nartist processing, which can significantly
increase the processing speed if you are only interested in album artist details.\n<br
/><br />\nPlease see the <a href="https://github.com/rdsdswift/picard-
plugins/blob/2.0_RDS_Plugins/plugins/additional_artists_details/docs/README.md"
>user\nguide</a> on GitHub for more information.</p>', 'files': {'__init__.py':
'38962f3ab630fffc173e1d4cf64edfee', 'docs/README.md':
'a6d9e21b8859b910406a62f9b0f362a7', 'docs/option_settings.png':
'f07ecfea99693b850424be0e3e954acb', 'options_additional_artists_details.ui':
'9889947eb555a6be7ecde40b6012615e', 'ui_options_additional_artists_details.py':
'1da9cf03278c6a5958211b9f73b1959a'}, 'license': 'GPL-2.0-or-later', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Additional Artists Details',
'version': '0.4'}, 'additional_artists_variables': {'api_versions': ['2.0', '2.1', '2.2', '2.7',
'2.9', '2.10', '2.11'], 'author': 'Bob Swift (rdsdswift)', 'description': '<p>This plugin
provides specialized album and track variables for use in\nnaming scripts. It is based
on the "Album Artist Extension" plugin, but\nexpands the functionality to also include
track artists. Note that it\ncannot be used as a direct drop-in replacement for the
"Album Artist\nExtension" plugin because the variables are provided with
different\nnames. This will require changes to existing scripts if switching to\nthis
plugin.\n<br /><br />\nPlease see the <a href="https://github.com/rdsdswift/picard-
plugins/blob/2.0_RDS_Plugins/plugins/additional_artists_variables/docs/README.m
d">user guide</a> on GitHub for more information.</p>', 'files':
{'additional_artists_variables.py': 'e3e8ce791fb39a1322aaf7a36c51f6e7'}, 'license':
'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name':
'Additional Artists Variables', 'version': '1.0'}, 'addrelease': {'api_versions': ['2.0'],
'author': 'Frederik "Freso" S. Olesen, Lukáš Lalinský, Philip Jägenstedt', 'description':
'<p>Adds a plugin context menu option to clusters and single files to help you quickly
add them as releases or standalone recordings to the MusicBrainz database via the
website by pre-populating artists, track names and times.</p>', 'files':
{'addrelease.py': '0fd3b1a02fb3199eee1fc77f6c56027d'}, 'name': 'Add Cluster As
Release', 'version': '0.7.3'}, 'albumartist_website': {'api_versions': ['2.0', '2.1', '2.2'],

```

'author': 'Sophist, Sambhav Kothari, Philipp Wolfer', 'description': "<p>Add's the album artist(s) Official Homepage(s)\n(if they are defined in the MusicBrainz database).</p>", 'files': {'albumartist_website.py': 'dbdbcd7c566dc5df528ee6a58c9b7376'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Album Artist Website', 'version': '1.2'}, 'albumartistextension': {'api_versions': ['2.0'], 'author': 'Bob Swift (rdsswift)', 'description': '<p>This plugin provides standardized, credited and sorted artist information\nfor the album artist. This is useful when your tagging or renaming scripts\nrequire both the standardized artist name and the credited artist name, or\nmore detailed information about the album artists.\n

\nThe information is provided in the following variables:</p>\n\n_aaeStdAlbumArtists = The standardized version of the album artists.\n_aaeCredAlbumArtists = The credited version of the album artists.\n_aaeSortAlbumArtists = The sorted version of the album artists.\n_aaeStdPrimaryAlbumArtist = The standardized version of the first\n(primary) album artist.\n_aaeCredPrimaryAlbumArtist = The credited version of the first (primary)\nalbum artist.\n_aaeSortPrimaryAlbumArtist = The sorted version of the first (primary)\nalbum artist.\n_aaeAlbumArtistCount = The number of artists comprising the album artist.\n\n<p>PLEASE NOTE: Once the plugin is installed, it automatically makes these \nvariables available to File Naming Scripts and other scripts in Picard. \nLike other variables, you must mention them in a script for them to affect \nthe file name or other data.\n

\nThis plugin is no longer being maintained. \nConsider switching to the \nAdditional Artists Variables plugin, \nwhich fills this \nrole, and also includes additional variables. That other plugin uses different \nnames for the album artist names provided here, so you if you switch plugins, you\nwill need to update your scripts with the different names.\n

\nVersion 0.6.1 of this plugin functions identically to Version 0.6. Only this \ndescription (and the version number) has changed.</p>', 'files': {'albumartistextension.py': '6382a36e2b8996b352cba63d8338e34a'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'AlbumArtist Extension', 'version': '0.6.1'}, 'amazon': {'api_versions': ['2.2', '2.3', '2.4', '2.5', '2.6', '2.7'], 'author': 'MusicBrainz Picard developers', 'description': '<p>Use cover art from Amazon.</p>', 'files': {'amazon.py': 'e5a4ecf562855fc57690ae1cfe7a486c'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Amazon cover art', 'version': '1.1'}, 'bpm': {'api_versions': ['2.0'], 'author': 'Len Joubert, Sambhav Kothari, Philipp Wolfer', 'description': '<p>Calculate BPM for selected files and albums. Linux only version with dependancy on Aubio and Numpy</p>', 'files': {'__init__.py': '6686bde897231c7ab9c8f53e3cd4b647', 'ui_options_bpm.py': '94c01295fe9ca23a87d40c9c8ec77c51', 'ui_options_bpm.ui': 'a8885df622a94580c9945c846634b364'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'BPM Analyzer', 'version': '1.5.2'}, 'classical_extras': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.5', '2.6', '2.7'], 'author': 'Mark Evens', 'description': '<p>Classical Extras provides tagging enhancements for Picard and, in particular,\nutilises MusicBrainz's hierarchy of works to provide

work/movement tags. All options are set through a user interface in Picard options->plugins. This interface provides separate sections into enhance artist/performer tags, works and parts, genres and also allows for a generalised "tag mapping" (simple scripting).

While it is designed to cater for the complexities of classical music tagging, it may also be useful for other music which has more than just basic song/artist/album data.

The options screen provides five tabs for users to control the tags produced:

1. Artists: Options as to whether artist tags will contain standard MB names, aliases or as-credited names. Ability to include and annotate names for specialist roles (chorus master, arranger, lyricist etc.). Ability to read lyrics tags on the file which has been loaded and assign them to track and album levels if required. (Note: Picard will not normally process incoming file tags).
2. Works and parts: The plugin will build a hierarchy of works and parts (e.g. Work -> Part -> Movement or Opera -> Act -> Number) based on the works in MusicBrainz's database. These can then be displayed in tags in a variety of ways according to user preferences. Furthermore partial recordings, medleys, arrangements and collections of works are all handled according to user choices. There is a processing overhead for this at present because MusicBrainz limits look-ups to one per second.
3. Genres etc.: Options are available to customise the source and display of information relating to genres, instruments, keys, work dates and periods. Additional capabilities are provided for users of Muso (or others who provide the relevant XML files) to use pre-existing databases of classical genres, classical composers and classical periods.
4. Tag mapping: in some ways, this is a simple substitute for some of Picard's scripting capability. The main advantage is that the plugin will remember what tag mapping you use for each release (or even track).
5. Advanced: Various options to control the detailed processing of the above.

All user options can be saved on a per-album (or even per-track) basis so that tweaks can be used to deal with inconsistencies in the MusicBrainz data (e.g. include English titles from the track listing where the MusicBrainz works are in the composer's language and/or script). Also existing file tags can be processed (not possible in native Picard).

See the readme file https://github.com/MetaTunes/picard-plugins/tree/metabrainz/2.0/plugins/classical_extras on GitHub here for full details.

```
{
  'files': {'Readme.md': 'aea0e89f69208b345a9b2590c02f3449',
    '__init__.py': '976182fc118a834b892cc09cf7e7214c', 'const.py':
    'b8bb158c518f175a9ee83e45ed134d02', 'options_classical_extras.ui':
    '7a89f51e49c64b73acdd6736bf04bce3', 'suffixtree.py':
    'e3e15409920e28eec753cf7c458cc5c3', 'ui_options_classical_extras.py':
    '7292d406ca481eb5f7f954695bbe1f20'}, 'license': 'GPL-2.0', 'license_url':
    'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Classical Extras', 'version':
    '2.0.14'}, 'classicdiscnumber': {'api_versions': ['0.15', '2.0'], 'author': 'Lukas Lalinsky',
    'description': '<p>Moves disc numbers and subtitles from the separate tags to album
    titles.</p>', 'files': {'classicdiscnumber.py': 'ec4abd529133d83ea0727db3d0e99699'},
    'name': 'Classic Disc Numbers', 'version': '0.2'}, 'collect_artists': {'api_versions': ['2.1',
    '2.2'], 'author': 'johbi', 'description': '<p>Adds a context menu shortcut to collect all
    track artists from a release and format them as the releases album artist.</p>', 'files':
```

{'collect_artists.py': 'adef9c3fcd5bcb4085bb493c45f7575'}, 'license': 'GPL-3.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl.txt', 'name': 'Collect Album Artists', 'version': '0.1'}, 'compatible_TXXX': {'api_versions': ['2.0'], 'author': 'Tungol', 'description': "<p>This plugin improves the compatibility of ID3 tags by using only a single value for TXXX frames. Multiple value TXXX frames technically don't comply with the ID3 specification.</p>", 'files': {'compatible_TXXX.py': 'd3dfadc913464a7d0df1daa81b929023'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Compatible TXXX frames', 'version': '0.1'}, 'critiquebrainz': {'api_versions': ['2.0'], 'author': 'Tobias Sarner', 'description': "<p>Uses Critiquebrainz for comment as review or rating.</p>\n<p>WARNING: Experimental plugin. All guarantees voided by use.</p>\n<p>Example: Taylor Swift\nRelease: Midnights\nhttps://musicbrainz.org/release/e348fdd6-f73b-47fe-94c4-670bfee26a39 ,\nhttps://critiquebrainz.org/release-group/0dcc84fb-c592-46e9-ba92-a52bb44dd553</p>\n<p>Recording:\nhttps://musicbrainz.org/recording/93113326-93e9-409c-a3d6-5ec91864ba30 ,\nhttps://critiquebrainz.org/recording/93113326-93e9-409c-a3d6-5ec91864ba30</p>", 'files': {'critiquebrainz.py': '78cbbdd3df33f0599f9e7a09e3fdf282'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Critiquebrainz Review Comment', 'version': '1.0.2'}, 'cuesheet': {'api_versions': ['2.0'], 'author': 'Lukáš Lalinský, Sambhav Kothari', 'description': "<p>Generate cuesheet (.cue file) from an album.</p>", 'files': {'cuesheet.py': '826c985a26c669f56cc0388cbebd3d6d'}, 'name': 'Generate Cuesheet', 'version': '1.2.2'}, 'decade': {'api_versions': ['2.0', '2.1', '2.2', '2.3'], 'author': 'Philipp Wolfer', 'description': "<p>Add a \$decade(date) function to get the decade from a year. E.g. \$decade(1994-04-05) will give \"90s\". By default decades between 1920 and 2000 will be shortened to two digits. You can disable this with setting the second parameter to 0, e.g. \$decade(1994,0) will give \"1990s\".</p>", 'files': {'__init__.py': 'd8e749fd80796e2265d8ecb5053ef44a'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Decade function', 'version': '1.0'}, 'decode_cyrillic': {'api_versions': ['1.0', '2.0'], 'author': 'aeontech', 'description': "<p>This plugin helps you quickly convert mis-encoded cyrillic Windows-1251 tags into proper UTF-8 encoded strings. If your track/album names look something like\n\"Àëèñà â ñòàíá ÷óääñ\", run this plugin from the context menu\nbefore running the \"Lookup\" or \"Scan\" tools</p>", 'files': {'decode_cyrillic.py': 'c590162021486d45fa10caa52200e15e'}, 'license': 'MIT', 'license_url': 'https://opensource.org/licenses/MIT', 'name': 'Decode Cyrillic', 'version': '1.1'}, 'decode_greek_cyrillic': {'api_versions': ['1.0', '2.0'], 'author': 'aeontech, Lefteris NeNpO', 'description': "<p>This plugin helps you quickly convert mis-encoded Greek Windows-1253 tags into proper UTF-8 encoded strings. If your track/album names look something like\n\"Àëèñà â ñòàíá ÷óääñ\", run this plugin from the context menu\nbefore running the \"Lookup\" or \"Scan\" tools</p>", 'files': {'decode_greek1253.py': 'fced06bcea2dbfdab6da68822ef3f56c'}, 'license': 'MIT', 'license_url': 'https://opensource.org/licenses/MIT', 'name': 'Decode Cyrillic Greek', 'version': '1.3'}, 'deezerart': {'api_versions': ['2.5'], 'author': 'Fabio Forni <livingsilver94>', 'description': "<p>Fetch cover arts from Deezer</p>", 'files': {'__init__.py': '5b35f31e2ed90b78133de41912869eae', 'deezer/__init__.py':

```

'f35ef270482c4cde5341b98c74612b9f', 'deezer/client.py':
'6128015576893bb397ea8842d3c56aa9', 'deezer/obj.py':
'a1e0555e20f9a25f14f7f4c6a85d062e', 'options.py':
'ba2c841825575a429639ea9aea36dc62', 'options.ui':
'55fb0f9c7b164c3d7561cb55b3cde32a'}, 'license': 'GPL-3.0-or-later', 'license_url':
'https://www.gnu.org/licenses/gpl-3.0.html', 'name': 'Deezer cover art', 'version':
'1.2.2'}, 'discnumber': {'api_versions': ['0.9.0', '0.10', '0.15', '2.0'], 'author': 'Lukas
Lalinsky', 'description': '<p>Moves disc numbers and subtitles from album titles to
separate tags. For example:<br/>\n<em>"Aerial (disc 1: A Sea of
Honey)"</em></p>\n<ul>\n <li>album = <em>"Aerial"</em></li>\n <li>discnumber =
<em>"1"</em></li>\n <li>discsubtitle = <em>"A Sea of Honey"</em></li>\n</ul>',
'files': {'discnumber.py': 'e5e63a013534bdb2be6c8b3899d3f336'}, 'name': 'Disc
Numbers', 'version': '0.1'}, 'enhanced_titles': {'api_versions': ['2.10'], 'author': 'Giorgio
Fontanive', 'description': '<p>This plugin sets the albumsort and titlesort tags. It also
provides the script\nfunctions $swapprefix_lang, $delprefix_lang and $title_lang. The
languages\nincluded at the moment are English, Spanish, Italian, German, French
and Portuguese.</p>\n<p>The functions do the same thing as their original
counterparts, but take\nmultiple languages as parameters. If no languages are
provided, all the available ones are\nincluded. Languages are provided with ISO
639-3 codes: eng, spa, ita, fra, deu, por.</p>\n<p>Tagging and checking aliases can
be disabled in the plugin\'s options page, found\nunder "plugins". Checking aliases
will slow down processing.</p>\n<p>If you wish to add your own language or to
change the words that are not capitalized,\nplease feel free to code the changes and
submit a pull request along with links to web\npages that give definitive language-
specific title-case rules.</p>', 'files': {'__init__.py':
'4cc257bbb46e88be5bc9d73aba777566', 'options_enhanced_titles.ui':
'894388a40adba0d6cc3e3348745629ad', 'ui_options_enhanced_titles.py':
'280f9fda0b0874af8ed9b1a70c35bc66'}, 'license': 'GPL-2.0', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Enhanced Titles', 'version': '0.1'},
'fanarttv': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.5', '2.6'], 'author': 'Philipp
Wolfer, Sambhav Kothari', 'description': '<p>Use cover art from fanart.tv.<br /><br
/>To use this plugin you have to register a personal API key on <a
href="https://fanart.tv/get-an-api-key/">fanart.tv</a>.</p>', 'files': {'__init__.py':
'2aa976cf73dd4e662cfb455377ec89d4', 'ui_options_fanarttv.py':
'6bde147d08f075c790413f12a00dcc90', 'ui_options_fanarttv.ui':
'59aba627fc950c0579f21770d1aacb76'}, 'license': 'GPL-2.0-or-later', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'fanart.tv cover art', 'version':
'1.6.3'}, 'featartist': {'api_versions': ['0.9.0', '0.10', '0.15', '0.16', '2.0'], 'author': 'Lukas
Lalinsky, Bryan Toth', 'description': '<p>Removes feat. artists from track titles.
Substitution is case insensitive.</p>', 'files': {'featartist.py':
'749c5d0f8259157e49db8753b490bdc3'}, 'name': 'Feat. Artists Removed', 'version':
'0.4'}, 'featartistsintitles': {'api_versions': ['0.9.0', '0.10', '0.15', '0.16', '2.0'], 'author':
'Lukas Lalinsky, Michael Wiencek, Bryan Toth, JeromyNix (NobahdiAtoll)',
'description': '<p>Move "feat." from artist names to album and track titles. Match is
case insensitive.</p>', 'files': {'featartistsintitles.py':
'4f935b4ba9e3eeb4dabc26867301ad77'}, 'name': 'Feat. Artists in Titles', 'version':

```

'0.5'}, 'fix_tracknums': {'api_versions': ['0.15', '1.0', '2.0'], 'author': 'Jonathan Bradley Whited', 'description': '<p>Fix the track numbers in a cluster by either using the track titles (1) or sequential order (2).</p>\n\n \n The title should contain something like "#-#" (number dash number) and be unique.
\n All non-numbers and non-dashes will be removed when comparing the titles.
\n This is especially useful for Language Audio Lessons, like this:\n <pre>- Title: "Unit 1 - Lesson 10"</pre>\n For example, take the following titles and track numbers:\n <pre>\n- Title: "Unit 1 - Lesson 1" - Track #1\n- Title: "Unit 1 - Lesson 2" - Track #1\n- Title: "Unit 2 - Lesson 10" - Track #2\n- Title: "Unit 2 - Lesson 1" - Track #2\n</pre>\n The track numbers will be changed to: 1, 2, 4, 3
\n The 3rd one will be changed to Track #4 because Lesson 1 < Lesson 10.
\n The titles will remain unchanged.\n \n\n \n The track numbers will be set based on the sequential order they appear within the cluster.\n\n\n<p>How to use:</p>\n\n \n Cluster a group of files\n \n Right click on the cluster\n \n Then click one:\n \n \n Plugins => Fix track numbers using titles\n \n Plugins => Fix track numbers using sequence\n \n \n', 'files': {'fix_tracknums.py': '665406e56ffa876f36caf36d4256a292'}, 'license': 'GPL-3.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl.txt', 'name': 'Fix Track Numbers', 'version': '0.2.1'}, 'format_performer_tags': {'api_versions': ['2.0'], 'author': 'Bob Swift, Philipp Wolfer', 'description': '<p>This plugin provides options with respect to the formatting of performer\ntags. It has been developed using the 'Standardise Performers' plugin by\nSophist as the basis for retrieving and processing the performer data for\neach of the tracks. The format of the resulting tags can be customized\nin the option settings page.</p>', 'files': {'__init__.py': '2c60a45606fefbb8473d7f5f2913208f', 'docs/HISTORY.md': '23336f12177011aa18533aafef224fd', 'docs/README.md': '2a70d2e870731a256c6ca6c7d99c1d05', 'docs/default_settings.jpg': '257f56b4f51157727f05d14244c92223', 'ui_options_format_performer_tags.py': '99f2c6abe70ec86708fdeb5ffd5d418e', 'ui_options_format_performer_tags.ui': '601e283ac1f08807c2501d605dfbeec0'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Format Performer Tags', 'version': '0.8.2'}, 'genre_mapper': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.6', '2.7', '2.8', '2.9', '2.10', '2.11'], 'author': 'Bob Swift', 'description': '<p>This plugin provides the ability to standardize genres in the "genre"\ntag by matching the genres as found to a standard genre as defined in\nthe genre replacement mapping configuration option. Once installed a\nsettings page will be added to Picard\'s options, which is where the\nplugin is configured.\n
\nPlease see the user guide on GitHub for more information.</p>', 'files': {'__init__.py': 'a881403723b98944c3e8ff6c3831058f', 'options_genre_mapper.ui': '59c6163104b2da85e4d149736e1827fd', 'ui_options_genre_mapper.py': '37d011c4113606747383a162a97a4777'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Genre Mapper', 'version': '0.6'}, 'haikuattrs': {'api_versions': ['2.2', '2.3', '2.4', '2.5', '2.6'], 'author': 'Philipp Wolfer', 'description': '<p>Save and load metadata to/from Haiku BFS attributes.</p>', 'files': {'haikuattrs.py': '875bc7c76e013736290cdbfa13aaa713'}, 'license': 'GPL-2.0-or-later',

'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Haiku BFS Attributes', 'version': '1.2'}, 'happidev_lyrics': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.5', '2.6'], 'author': 'Andrea Avallone, Philipp Wolfer', 'description': '<p>Fetch lyrics from Happi.dev Lyrics, which provides millions of lyrics from artist all around the world. Lyrics provided are for educational purposes and personal use only. Commercial use is not allowed.

In order to use Happi.dev you need to get a free API key at happi.dev</p>', 'files': {'happidev_lyrics.py': '48fa5b46c22962e0c3289f912c9af762'}, 'license': 'MIT', 'license_url': 'https://opensource.org/licenses/MIT', 'name': 'Happi.dev Lyrics', 'version': '2.1.2'}, 'hyphen_unicode': {'api_versions': ['0.9', '0.10', '0.11', '0.15', '2.0'], 'author': 'Alan Swanson <revier@improbability.net>', 'description': '<p>Replaces unicode character HYPHEN (U+2010) [0xE2 0x80 0x90] with typographically identical HYPHEN-MINUS (U+002D) [0x2D] for fonts\ntthat do not support HYPHEN and to prevent visually duplicate filenames\ndifferentiated only by their hyphens.</p>\n<p>Unicode duplicated hyphen from ASCII as an unambiguous way to designate a\ntyphen from a minus whilst still being typographically indential. Since\ntext processing on music tags is rare so choice is purely pedantic esepcially\nas keyboards only have HYPHEN-MINUS.</p>\n<p>Replaces character on "album", "title", "artist", "artists", "artistsort",\n"albumartist", "albumartists" and "albumartistsort" tags.</p>', 'files': {'hyphen_unicode.py': 'b224d88f1c77b8e80093a4da2ee24a3e'}, 'license': 'GPL-3.0-or-later', 'license_url': 'https://gnu.org/licenses/gpl.html', 'name': 'Hyphen unicode', 'version': '1.0.1'}, 'instruments': {'api_versions': ['2.0'], 'author': 'David Mandelberg', 'description': '<p>Adds a multi-valued tag (~instruments) containing all the instruments (including vocals), \nfor use in scripts.</p>', 'files': {'instruments.py': 'd97960ce7f31682981e58940a59b9196'}, 'license': 'GPL-3.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-3.0.html', 'name': 'Instruments', 'version': '1.0.1'}, 'keep': {'api_versions': ['0.15.0', '0.15.1', '0.16.0', '1.0.0', '1.1.0', '1.2.0', '1.3.0', '2.0'], 'author': 'Wieland Hoffmann', 'description': '<p>Adds a \$keep() function to delete all tags except the ones that you want.\nTags beginning with <code>musicbrainz_</code> are kept automatically, as are tags\nbeginning with <code>_</code>.</p>\n<p>To keep all tags that can have a description (like <code>comment</code>, lyrics<code>and</code>performer<code>), add<code><tagname without description><code>(not including</code>:) to the\nlist of tags to keep.</p>', 'files': {'keep.py': 'a8a6b8448cb3f358c07a983842ab9378'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Keep tags', 'version': '1.2.1'}, 'key_wheel_converter': {'api_versions': ['2.3', '2.4', '2.6', '2.7', '2.13'], 'author': 'Bob Swift', 'description': '<p>Adds functions to convert between 'standard', 'camelot', 'open key' and 'traktor' key formats.</p>', 'files': {'key_wheel_converter.py': 'b5a96af2ca27f0f06c0114b7b41e2ea6'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Key Wheel Converter', 'version': '1.2'}, 'lastfm': {'api_versions': ['2.0'], 'author': 'Lukáš Lalinský, Philipp Wolfer', 'description': '<p>Use tags from Last.fm as genre.</p>', 'files': {'__init__.py': '329be6b67153bf9b3cfc1fbc91a56c29', 'ui_options_lastfm.py': '3a3a19ba1116645cf4a2d01f8e454a10', 'ui_options_lastfm.ui':

'0cd776c4ce19cbeccdc6b508d63ea93e'}, 'name': 'Last.fm', 'version': '0.10.1'},
'loadasnat': {'api_versions': ['1.4.0', '2.0', '2.1', '2.2'], 'author': 'Philipp Wolfer',
'description': '<p>Allows loading selected tracks as non-album tracks. Useful for
tagging single tracks where you do not care about the album.</p>', 'files':
{'loadasnat.py': 'cc1bad0a7c0f7a9aef730470c96f0a46'}, 'license': 'GPL-2.0-or-later',
'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Load as non-album
track', 'version': '0.4'}, 'losslessfuncs': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.5',
'2.6', '2.7', '2.8', '2.9', '2.10', '2.11'], 'author': 'Philipp Wolfer', 'description': '<p>Tagger
script functions to detect if a file is lossless or lossy</p>', 'files': {'__init__.py':
'22523f93104feffbe43cc9360b17f7ca'}, 'license': 'GPL-2.0', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Tagger script functions
\$is_lossless() and \$is_lossy()', 'version': '0.3'}, 'lrclib_lyrics': {'api_versions': ['2.12'],
'author': 'Giorgio Fontanive', 'description': '<p>Fetches lyrics from
Lrclib.net</p>\n<p>Also allows to export lyrics to an .lrc file or import them from
one.</p>', 'files': {'__init__.py': '35e0d32a2cd66d4e9b685fea95af2b76',
'option_lrclib_lyrics.py': 'f7d0fb3e48a0232f34a2182a722bd8be',
'option_lrclib_lyrics.ui': 'e6ad31842339ae83153e4d2b966c5b35'}, 'license': 'GPL-
2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Lrclib Lyrics',
'version': '0.1'}, 'mod': {'api_versions': ['2.8'], 'author': 'Philipp Wolfer', 'description':
'<p>Support for loading and renaming various tracker module formats (.mod, .xm, .it,
.mptm, .ahx, .mtm, .med, .s3m, .ult, .699, .okt). There is limited support for writing the
title tag as track name for some formats.</p>', 'files': {'__init__.py':
'b4f5d11a5e25e882ab5bc5573c92a71b'}, 'license': 'GPL-2.0', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'MOD files', 'version': '0.2.1'},
'moodbars': {'api_versions': ['2.0'], 'author': 'Len Joubert, Sambhav Kothari',
'description': '<p>Calculate Moodbars for selected files and albums.

\nAccording to WikiPedia\nna
"Moodbar is a computer visualization used for navigating within a piece of music or
any other recording on a digital audio track.\nThis is done with a commonly
horizontal bar that is divided into vertical stripes.\nEach stripe has a colour
combination showing the "mood" within a short part of the audio track."

\nTo use this plugin you will need to download special executables to create the
moodbars -\nat the time of writing, executables are only available for various Linux
distributions\n(see the Amarok Moodbar
page<a> for details).</p>', 'files': {'__init__.py':
'f07c72ff9d9e5a8923a30411681681b8', 'ui_options_moodbar.py':
'ca7b615b705d0a646075e7ecc856355d', 'ui_options_moodbar.ui':
'710fa292a99225b4cfb9fb9661382428'}, 'license': 'GPL-2.0', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Moodbars', 'version': '2.3.3'},
'musixmatch': {'api_versions': ['2.0'], 'author': 'm-yn, Sambhav Kothari, Philipp
Wolfer', 'description': '<p>Fetch first 30% of lyrics from Musixmatch</p>', 'files':
{'README': '8ead2860a1e939898c789f86f51dc7e3', '__init__.py':
'c79d43947409fe168d9c513ec58d2789', 'ui_options_musixmatch.py':
'fae79dc04af8f302d6a25aab4c1424be'}, 'license': 'GPL-2.0', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Musixmatch Lyrics', 'version':

'1.1.1'}, 'no_release': {'api_versions': ['2.0'], 'author': 'Johannes Weißl, Philipp Wolfer', 'description': '<p>Do not store specific release information in releases of unknown origin.</p>', 'files': {'no_release.py': '6e46429c3604cd4f1d66387d2176120a'}, 'name': 'No release', 'version': '0.3'}, 'non_ascii_equivalents': {'api_versions': ['0.9', '0.10', '0.11', '0.15', '2.0'], 'author': 'Anderson Mesquita <andersonvom@trysometinghere>, Konrad Marciniak', 'description': '<p>Replaces accented and otherwise non-ASCII characters\nwith a somewhat equivalent version of their ASCII counterparts. This allows old\ndevices to be able to display song artists and titles somewhat correctly,\ninstead of displaying weird or blank symbols. It\'s an attempt to do a little\nbetter than Musicbrainz\'s native "Replace non-ASCII characters" option.</p>\n<p>Currently replaces characters on "album", "albumartist", "albumartists", "albumartistsort", "albumsort", "artist", "artists", "artistsort" and "title" tags.</p>', 'files': {'non_ascii_equivalents.py': 'fe95ba26afec1e1b004ade9412c1da4b'}, 'license': 'GPL-3.0-or-later', 'license_url': 'https://gnu.org/licenses/gpl.html', 'name': 'Non-ASCII Equivalents', 'version': '0.5'}, 'padded': {'api_versions': ['0.15.0', '0.15.1', '0.16.0', '1.0.0', '1.1.0', '1.2.0', '1.3.0', '2.0'], 'author': 'Wieland Hoffmann', 'description': '<p>Adds padded disc- and tracknumbers so the length of all disc- and tracknumbers\nis the same. They are stored in the <code>_paddedtracknumber</code> and <code>_paddeddiscnumber</code>\ntags.</p>', 'files': {'padded.py': '4f40bf3c25d96e97d26533a77d66aab0'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Padded disc and tracknumbers', 'version': '1.0.1'}, 'papercdcase': {'api_versions': ['2.0', '2.1', '2.2'], 'author': 'Philipp Wolfer, Sambhav Kothari', 'description': '<p>Create a paper CD case from an album or cluster using http://papercdcase.com</p>', 'files': {'papercdcase.py': 'cd343275cf0cf20ea6a2303d03b43e59'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Paper CD case', 'version': '1.2.1'}, 'performer_tag_replace': {'api_versions': ['2.0'], 'author': 'Bob Swift (rdswift)', 'description': '<p>This plugin provides the ability to replace text in performer tags. It\nhas been developed using the \'Standardise Performers\' plugin by Sophist\nas the basis for retrieving and processing the performer data for each\nof the tracks. The original/replacement pairs used can be customized\nin the option settings page.\n

\nPlease see the user\nguide on GitHub for more information.</p>', 'files': {'__init__.py': '4a66c79514f7092fad231765b43ae62d', 'options_performer_tag_replace.ui': '87e88fb261b11f6f674f7e354a50ea1b', 'ui_options_performer_tag_replace.py': 'f438747e8bb2ff81e93875766342dad8'}, 'license': 'GPL-2.0 or later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Performer Tag Replace', 'version': '0.03'}, 'persistent_variables': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.6', '2.7'], 'author': 'Bob Swift (rdswift)', 'description': '<p>\nThis plugin provides the ability to store and retrieve script variables that persist across tracks and albums.\nThis allows things like finding and storing the earliest recording date of all of the tracks on an album.\n</p>\n<p>\nThere are two types of persistent variables maintained - album variables and session variables. Album variables\npersist across all tracks on an album. Each album\'s information is stored separately, and is reset

when the album is refreshed. The information is cleared when an album is removed. Session variables persist across all albums and tracks, and are cleared when Picard is shut down or restarted.

This plugin adds eight new scripting functions to allow management of persistent script variables:

- `$set_a(name,value)` : Sets the album persistent variable name to value.
- `$unset_a(name)` : Unsets the album persistent variable name.
- `$get_a(name)` : Gets the album persistent variable name.
- `$clear_a()` : Clears all album persistent variables.
- `$set_s(name,value)` : Sets the session persistent variable name to value.
- `$unset_s(name)` : Unsets the session persistent variable name.
- `$get_s(name)` : Gets the session persistent variable name.
- `$clear_s()` : Clears all session persistent variables.

Please see the https://github.com/rdswift/picard-plugins/blob/2.0_RDS_Plugins/plugins/persistent_variables/docs/README.md user guide on GitHub for more information.

'files': {'__init__.py': '5d0144c8d0ea2d4a977d5dae9fbc952d', 'ui_variables_dialog.py': '1570a42b07938d50f6a1583be5e0c886'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Persistent Variables', 'version': '1.1'}, 'playlist': {'api_versions': ['2.0'], 'author': 'Francis Chin, Sambhav Kothari, Chris Hylen', 'description': '<p>Generate an Extended M3U playlist (.m3u8 file, UTF8\\nencoded text). Relative pathnames are used where audio files are in the same\\ndirectory as the playlist, otherwise absolute (full) pathnames are used.</p>', 'files': {'playlist.py': '4c6c99917d7ef558fb061332ea2eff3a'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Generate M3U playlist', 'version': '1.2.1'}, 'post_tagging_actions': {'api_versions': ['2.10', '2.11'], 'author': 'Giorgio Fontanive', 'description': '<p>This plugin lets you set up actions that run with a context menu click. \\nAn action consists in a command line executed for each album or each track along\\nwith a few options to tweak the behaviour. \\nThis can be used to run external programs and pass some variables to it. </p>', 'files': {'__init__.py': '510a0ac4facddb6550cb5ec70f9f5f7', 'actions_status.py': '65d59f32e043e814c491a3046c6941fc', 'actions_status.ui': 'ec999f34323522933569e91f5bd86b99', 'docs/guide.md': 'fbc4af3e8a666f6fdbdcf0011a5a5978', 'docs/options.png': '6f89c36c6203c9b315a53cd01fe43834', 'options_post_tagging_actions.py': '56ac5f6dc0017cd11482b09da2ecf4a5', 'options_post_tagging_actions.ui': '1eca69c5cc884f92adf8173d1b401335'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Post Tagging Actions', 'version': '0.1'}, 'release_type': {'api_versions': ['0.9.0', '0.10', '0.15', '2.0'], 'author': 'Elliot Chance', 'description': '<p>Appends information to EPs and Singles</p>', 'files': {'release_type.py': 'c66f0989f9ebcc94bc3ed183806e8b91'}, 'name': 'Release Type', 'version': '1.4'}, 'releasetag_aggregations': {'api_versions': ['2.5', '2.6'], 'author': 'Philipp Wolfer', 'description': '<p>Add functions to aggregate tags on a release:</p>\$album_all(name)\\u200e\$album_avg(name, precision=2)\\u200e\$album_max(name, precision=2)\\u200e\$album_min(name,

precision=2)\u200e\$album_mode(name)\u200e\$album_distinct(name, separator=;)\u200e\$album_multi_avg(name, precision=2)\u200e\$album_multi_max(name, precision=2)\u200e\$album_multi_min(name, precision=2)\u200e\$album_multi_mode(name)\u200e\$album_multi_distinct(name, separator=;)The functions work only in file naming

scripts and the files should either be part of a release or cluster!</p>', 'files': {'releasetag_aggregations.py': '82805070b9373021f3a62aa81780750c'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Release tag aggregation functions', 'version': '0.4'}, 'remove_perfect_albums': {'api_versions': ['2.0', '2.1', '2.2', '2.3'], 'author': 'ichneumon, hrglgrmpf', 'description': '<p>Remove all perfectly matched albums from the selection.</p>', 'files':

{'remove_perfect_albums.py': '46d3a77e17b676cce4b329ebf828d5b3'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Remove Perfect Albums', 'version': '0.3'}, 'reorder_sides': {'api_versions': ['2.0'], 'author': 'David Mandelberg, Sambhav Kothari', 'description': '<p>Split mediums and re-order sides to match side order rather than\n medium order. E.g., if a release has two mediums with track numbers\n A1, A2, ..., D1, D2, ... and B1, B2, ..., C1, C2,\n ..., this plugin will split the release into four mediums and\n reorder the new mediums so that the track numbers are A1, A2,\n ..., B1, B2, ..., C1, C2, ..., D1, D2, ...</p>\n<p>This is primarily intended to make vinyl records designed for record\n changers\n

(https://en.wikipedia.org/wiki/Record_changer#Automatic_sequencing)\n play in the correct order.</p>', 'files': {'reorder_sides.py':

'5306e70b65db0b8e2892c30133b7385b'}, 'license': 'GPL-3.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-3.0.html', 'name': 'Re-order sides of a release', 'version': '1.2'}, 'replace_forbidden_symbols': {'api_versions': ['0.9', '0.10', '0.11', '0.15', '2.0', '2.2'], 'author': 'Alex Rustler <alex_rustler@rambler.ru>', 'description': '<p>Replaces Windows forbidden symbols: :, /, *, ?, ", ., | etc.\n with a similar UNICODE version.\n Currently replaces characters on "album", "artist",\n "title", "albumartist", "releasetype", "label" tags.\n Also add \$replace_forbidden() function for Tagger.\n Example: \$set(composer,\$script_forbidden(%composer%))</p>', 'files': {'replace_forbidden_symbols.py': '5af9047e3c336ab95be09ee85090190e'}, 'license': 'GPL-3.0-or-later', 'license_url': 'https://gnu.org/licenses/gpl.html', 'name': 'Replace Forbidden Symbols', 'version': '0.3'}, 'replaygain2': {'api_versions': ['2.0'], 'author': 'complexlogic', 'description': '<p>Calculates ReplayGain information for tracks and albums according to the\n<a

[href="https://wiki.hydrogenaud.io/index.php?title=ReplayGain_2.0_specification">ReplayGain 2.0 specification.\n This plugin depends on the ReplayGain utility rsgain. Users\n are required to install rsgain and set its path in the plugin settings before](https://wiki.hydrogenaud.io/index.php?title=ReplayGain_2.0_specification)

use.</p>\n<h4>Usage</h4>\n<p>Select one or more tracks, albums, or clusters then right click and select Plugin->Calculate ReplayGain.\n The plugin will calculate ReplayGain information for the selected items and display the results in the\n metadata window. Click the save button to write the tags to file.</p>\n<p>The following file formats are supported:</p>\n\nMP3 (.mp3)\nFLAC

(.flac)\nOgg (.ogg, .oga, spx)\nOpus (.opus)\nMPEG-4 Audio (.m4a, .mp4)\nWavpack (.wv)\nMonkey's Audio (.ape)\nWMA (.wma)\nMP2 (.mp2)\nWAV (.wav)\nAIFF (.aiff)\nTAK (.tak)\nMusePack (Stream Version 8 only) (.mpc)\n\n<p>This plugin is based on the original ReplayGain plugin by Philipp Wolfer and Sophist.</p>
'files': {'__init__.py': 'f67bc13f69c159ebb70215349052b06f', 'ui_options_replaygain2.py': '601c54bfd09f908f0b626698b5a46432', 'ui_options_replaygain2.ui': 'b45c5ba5c5e609eec0b81362fd7c268a'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'ReplayGain 2.0', 'version': '1.8'},
'save_and_rewrite_header': {'api_versions': ['0.9.0', '0.10', '0.15', '2.0'], 'author': 'Nicolas Cenerario', 'description': '<p>This plugin adds a context menu action to save files and rewrite their header.</p>', 'files': {'save_and_rewrite_header.py': '18375c5625822596c1df0a769c220288'}, 'license': 'GPL-3.0-or-later', 'license_url': 'http://www.gnu.org/licenses/gpl-3.0.txt', 'name': 'Save and rewrite header', 'version': '0.3'},
'script_logger': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.6', '2.9'], 'author': 'Bob Swift (rdswift)', 'description': "<p>This plugin provides a new script function <code>\$logline()</code> to write entries\nto Picard's system log. By default, the log level is set at <code>Info</code>, but\nany level can be used by providing the level as an optional second\nparameter to the function.</p>\n<p>The function is used as:</p>\n<p><code>\$logline(text[,level])</code></p>\n<p>where <code>text</code> is the text to write to the log. The entry will be written\nat log level <code>Info</code> by default, but this can be changed by specifying a\ndifferent level as an optional second parameter. Allowable log levels are:</p>\n\nE (Error)\nW (Warning)\nI (Info)\nD (Debug)\n\n<p>If an unknown level is entered, the function will use the default <code>Info</code>\nlevel.</p>", 'files': {'__init__.py': '58682cc4c596eea9f1500b18281092e3'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Script Logger', 'version': '0.1'},
'search_engine_lookup': {'api_versions': ['2.0', '2.1', '2.2', '2.3'], 'author': 'Bob Swift', 'description': '<p>Adds a right click option on a cluster to look up album information using a search engine in a browser window.</p>\n<p>Adds a right click option on an album or track to look up cover art for the selected album or title.</p>', 'files': {'README.md': '2b73f5c4acc3cb90bfd84d13954d0638', '__init__.py': '7547e3fc3ff0c9441afe53b01c310291', 'ui_options_search_engine_editor.py': 'afdd9fa0eaf8e13bcc982955e9629fa9', 'ui_options_search_engine_editor.ui': 'affbb6cd2adaa9f0d1d6533327ef8676', 'ui_options_search_engine_lookup.py': '0480990c911854b3c9af0a2eb3e93da1', 'ui_options_search_engine_lookup.ui': '3127bac9fd30ec58a9774cc6b0b12c53'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Search Engine Lookup', 'version': '2.1.0'},
'smart_title_case': {'api_versions': ['2.0'], 'author': 'Sophist based on an earlier plugin by Javier Kohen', 'description': '<p>Capitalize First Character In Every Word Of Album/Track Title/Artist.
\nLeaves words containing embedded uppercase as-is i.e. USA or DoA.
\nFor Artist/AlbumArtist, title cases only artists not join phrases
\ne.g. The Beatles feat. The Who.</p>', 'files': {'smart_title_case.py': '5851c22ffe01141fe8e7e7b5771463c8'}, 'license': 'GPL-2.0-or-

later', 'license_url': 'https://www.gnu.org/licenses/gpl-3.0.html', 'name': 'Smart Title Case', 'version': '0.4.2'}, 'sort_multivalue_tags': {'api_versions': ['0.15', '2.0'], 'author': 'Sophist', 'description': '<p>This plugin sorts multi-value tags e.g. Performers alphabetically.

\nNote: Some multi-value tags are excluded for the following reasons:</p>\n\nSequence is important e.g. Artists\nThe sequence of one tag is linked to the sequence of another e.g. Label and Catalogue number.\n', 'files': {'sort_multivalue_tags.py': 'e8eee87b9cca38600cc635afeb7cce26'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Sort Multi-Value Tags', 'version': '1.1'}, 'soundtrack': {'api_versions': ['1.0', '2.0'], 'author': 'Samir Benmendil', 'description': '<p>Sets the albumartist to "Soundtrack" if releasetype is a soundtrack.</p>', 'files': {'soundtrack.py': '86606046dc6144a4ad9f039e4c6f8487'}, 'license': 'WTFPL', 'license_url': 'http://www.wtfpl.net/', 'name': 'Soundtrack', 'version': '0.2'}, 'standardise_feat': {'api_versions': ['1.4', '2.0', '2.1', '2.2', '2.3'], 'author': 'Sambhav Kothari', 'description': '<p>Standardises "featuring" join phrases for artists to "feat."</p>', 'files': {'standardise_feat.py': 'c00cb5bb9a9a82a083fd7ddccb65b372'}, 'license': 'GPL-3.0', 'license_url': 'http://www.gnu.org/licenses/gpl-3.0.txt', 'name': 'Standardise Feat.', 'version': '0.3'}, 'standardise_performers': {'api_versions': ['2.0'], 'author': 'Sophist', 'description': '<p>Splits multi-instrument performer tags into single\ninstruments and combines names so e.g. (from 10cc by 10cc track 1):</p>\n<pre>\nPerformer [acoustic guitar, bass, dobro, electric guitar and tambourine]: Graham Gouldman\nPerformer [acoustic guitar, electric guitar, grand piano and synthesizer]: Lol Creme\nPerformer [electric guitar, moog and slide guitar]: Eric Stewart\n</pre>\n<p>becomes:</p>\n<pre>\nPerformer [acoustic guitar]: Graham Gouldman;\nPerformer [bass]: Graham Gouldman\nPerformer [dobro]: Graham Gouldman\nPerformer [electric guitar]: Eric Stewart;\nPerformer [grand piano]: Lol Creme\nPerformer [moog]: Eric Stewart\nPerformer [slide guitar]: Eric Stewart\nPerformer [synthesizer]: Lol Creme\nPerformer [tambourine]: Graham Gouldman\n</pre>\n<p>Update: This version now sorts the performer tags in order to maintain a consistent value and avoid tags appearing to change even though the base data is equivalent.</p>', 'files': {'standardise_performers.py': '5b972b4dd81b352eeb1de072d6912641'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Standardise Performers', 'version': '1.0'}, 'submit_folksonomy_tags': {'api_versions': ['2.2', '2.9'], 'author': 'Flaky', 'description': '<p>A plugin allowing submission of specific tags on tracks you own (defaults to <i>genre</i> and <i>mood</i>) as folksonomy tags on MusicBrainz. Supports submitting to recording, release, release group and release artist entities.</p>\n<p>A MusicBrainz login is required to use this plugin. Log in first by going to the General options. Then, to use, right click on a track or release then go to <i>Plugins</i> and depending on what you want to submit, choose the option you want.</p>\n<p>Uses code from rdswift's "Submit ISRC" plugin (specifically, the handling of the network response)</p>', 'files': {'README.md': '1c3fcdabc7ee5fe6c6aeb13ed7582a81', '__init__.py': '1b9694e33b8bd4354e7ab3a4073500bf', 'ui_config.py': '5b910e78b7f7fdc80e02527bd9ae86a2'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Submit Folksonomy Tags',

'version': '0.3'}, 'submit_isrc': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.6', '2.9'], 'author': 'Bob Swift', 'description': '<p>\nAdds a right click option on an album to submit the ISRCs to the MusicBrainz server\nspecified in the Options.\n</p>\n<p>\nTo use this function, you must first match your files to the appropriate tracks for\na release. Once this is done, but before you save your files if you have Picard set\nto overwrite the \'isrc\' tag in your files, right-click the release and select "Submit\nISRCs" in the "Plugins" section. For each file that has a single valid ISRC in its\nmetadata, the ISRC will be added to the recording on the release if it does not\nalready exist. Once all tracks for the release have been processed, the missing\nISRCs will be submitted to MusicBrainz.\n</p>\n<p>\nIf a file\'s metadata contains multiple ISRCs, such as if the file has already been\ntagged, then no ISRCs will be submitted for that file.\n</p>\n<p>\nIf one of the files contains an invalid ISRC, or if the same ISRC appears in the\nmetadata for two or more files, then a notice will be displayed and the submission\nprocess will be aborted.\n</p>\n<p>\nWhen ISRCs have been submitted, a notice will be displayed showing whether or not\nthe submission was successful.\n</p>', 'files': {'README.md': '468166e4537e97efabc34ab552aa67f5', '__init__.py': 'c03990caf114e92ff6df8f5517ebb86a'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Submit ISRC', 'version': '1.1'}, 'tangoinfo': {'api_versions': ['2.6', '2.7'], 'author': 'Felix Elsner, Sambhav Kothari, Philipp Wolfer', 'description': '<p>Load genre, date and vocalist tags for latin dance music\nfrom tango.info.\n</p>', 'files': {'README.md': '721e8cb08e8b4ae69f66187157063bd5', '__init__.py': '2c7a54bd0f2bd8bf23edf35b7db82309'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Tango.info Adapter', 'version': '0.2.0'}, 'theaudiodb': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.5', '2.6'], 'author': 'Philipp Wolfer', 'description': '<p>Use cover art from TheAudioDB.\n</p>', 'files': {'__init__.py': '86cf840d7f9302d61cbda7edf011f4f5', 'ui_options_theaudiodb.py': 'c19d8c4d8a9eeddfb29792b314f204dd', 'ui_options_theaudiodb.ui': '57b6048ba8166b0a460322466e713454'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'TheAudioDB cover art', 'version': '1.3.1'}, 'titlecase': {'api_versions': ['2.0'], 'author': 'Javier Kohen, Sambhav Kothari', 'description': '<p>Capitalize First Character In Every Word Of A Title\n</p>', 'files': {'titlecase.py': '0da64f5950dbc5c2ae3c5b016e428e47'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Title Case', 'version': '1.0.2'}, 'tracks2clipboard': {'api_versions': ['2.0'], 'author': 'Michael Elsdörfer, Sambhav Kothari', 'description': '<p>Exports a cluster\'s tracks to the clipboard, so it can be copied into the tracklist field on MusicBrainz\n</p>', 'files': {'tracks2clipboard.py': 'c9efa9499819980dcfc5f565ce6a2108'}, 'name': 'Copy Cluster to Clipboard', 'version': '1.0'}, 'viewvariables': {'api_versions': ['2.0'], 'author': 'Sophist', 'description': '<p>Display a dialog box listing the metadata variables for the track / file.\n
\nThis allows you to see metadata variables beginning with "~" which are not normally visible in the metadata\npane of the main Picard window, which can be useful when you are writing tagging or file naming scripts.\n</p>', 'files': {'__init__.py': '0ffaa4f0b7989c71953d7a49f24f5a0b', 'ui_variables_dialog.py': 'e9f8b8e57c6785054b6188223e7710df', 'ui_variables_dialog.ui':

```
'f67e419bf358d920a68a8cc964b062fd'}, 'license': 'GPL-2.0', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'View script variables', 'version':
'0.7.2'}, 'wikidata': {'api_versions': ['2.0', '2.1', '2.2'], 'author': 'Daniel Sobey, Sambhav
Kothari', 'description': '<p>Query wikidata to get genre tags</p>', 'files': {'__init__.py':
'45d3410e3aa3392c8333345b85f1c78f', 'ui_options_wikidata.py':
'71e408c2e3922e66bbaa4c54f936bd49', 'ui_options_wikidata.ui':
'f1eb5acfc21a654d9726a19c28e0bd35'}, 'license': 'WTFPL', 'license_url':
'http://www.wtfpl.net/', 'name': 'Wikidata Genre', 'version': '1.4.5'},
'workandmovement': {'api_versions': ['2.1', '2.2', '2.3', '2.4', '2.5', '2.6', '2.7', '2.8'],
'author': 'Philipp Wolfer', 'description': '<p>Set work and movement based on work
relationships.</p>\n<p>This plugin attempts to set the <code>movement</code>
and <code>movementnumber</code> tags, but only\nif the work linked to the
recording is part of a larger work. The <code>work</code> tag\nthen gets set to the
work of which the movement is a part of.</p>\n<p>If the recording is only linked to a
simple work without separate parts then it\nis not considered a proper work and the
work and movement related tags will be\ncleared.</p>\n<p>The plugin will always
set the original values, as loaded from MusicBrainz, of\nthe <code>work</code> and
<code>musicbrainz_workid</code> tags into the variables
<code>%%recording_work%</code>\nand <code>%%recording_workid%</code> to
be used in scripting.</p>', 'files': {'__init__.py':
'52074ee438ed26e925146d00bd0f6f30', 'roman.py':
'a0f22ff96d6e08bef57e8dfa5a451807'}, 'license': 'GPL-2.0-or-later', 'license_url':
'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Work & Movement', 'version':
'1.1'}}}
```

```
D: 21:13:14,865 webservice/ratecontrol._out_of_backoff:231:
('picard.musicbrainz.org', 443): oobackoff; delay: 1000ms -> 1000ms; slow start;
window size 1.000 -> 2.000
```