D: 16:23:35,075 tagger.__init__:315: Starting Picard from '/Applications/MusicBrainz Picard.app/Contents/MacOS/picard/tagger.pyc'
D: 16:23:35,076 tagger.__init__:316: Platform: macOS-11.7.10-x86_64-i386-64bit CPython 3.11.5
D: 16:23:35,076 tagger.__init__:318: Versions: Picard 2.11.0rc1, Python 3.11.5, PyQt 5.15.10, Qt 5.15.12, Mutagen 1.47.0, Discid discid 1.2.0, libdiscid 0.6.4, astrcmp C, SSL Secure Transport, macOS 11.7
D: 16:23:35,076 tagger.__init__:319: Configuration file path: '/Users/Nova/.config/MusicBrainz/Picard.ini'
D: 16:23:35,076 tagger.__init__:321: User directory: '/Users/Nova/Library/Preferences/MusicBrainz/Picard'
D: 16:23:35,076 tagger.__init__:322: System long path support: True
D: 16:23:35,077 tagger.__init__:325: Qt Env.: QT_PLUGIN_PATH='/Applications/MusicBrainz Picard.app/Contents/MacOS/PyQt5/Qt5/plugins'
D: 16:23:35,077 i18n.setup_gettext:146: UI language: system
D: 16:23:35,077 i18n._log_lang_env_vars:131: Env vars:
D: 16:23:35,081 i18n.setup_gettext:154: Trying locales: ['en_US']
D: 16:23:35,089 i18n.setup_gettext:160: Set locale to: 'en_US'
D: 16:23:35,089 i18n.setup_gettext:171: Using locale: 'en_US'
D: 16:23:35,091 i18n._load_translation:118: Loading gettext translation for picard, localedir='/Applications/MusicBrainz Picard.app/Contents/MacOS/locale', language='en_US'
D: 16:23:35,091 i18n._load_translation:121: [Errno 2] No translation file found for domain: 'picard'
D: 16:23:35,092 i18n._load_translation:118: Loading gettext translation for picard-attributes, localedir='/Applications/MusicBrainz Picard.app/Contents/MacOS/locale', language='en_US'
D: 16:23:35,092 i18n._load_translation:121: [Errno 2] No translation file found for domain: 'picard-attributes'
D: 16:23:35,092 i18n._load_translation:118: Loading gettext translation for picard-constants, localedir='/Applications/MusicBrainz Picard.app/Contents/MacOS/locale', language='en_US'
D: 16:23:35,093 i18n._load_translation:121: [Errno 2] No translation file found for domain: 'picard-constants'
D: 16:23:35,093 i18n._load_translation:118: Loading gettext translation for picard-countries, localedir='/Applications/MusicBrainz Picard.app/Contents/MacOS/locale', language='en_US'
D: 16:23:35,093 i18n._load_translation:121: [Errno 2] No translation file found for domain: 'picard-countries'
D: 16:23:35,093 i18n.setup_gettext:191: _ = <bound method NullTranslations.gettext of <gettext.NullTranslations object at 0x10f6ce9d0>>
D: 16:23:35,094 i18n.setup_gettext:192: N_ = <function <lambda> at 0x10d590c20>
D: 16:23:35,094 i18n.setup_gettext:193: ngettext = <bound method NullTranslations.ngettext of <gettext.NullTranslations object at 0x10f6ce9d0>>
D: 16:23:35,094 i18n.setup_gettext:194: gettext_countries = <bound method NullTranslations.gettext of <gettext.NullTranslations object at 0x10f6ce8d0>>
D: 16:23:35,094 i18n.setup_gettext:195: gettext_attributes = <bound method NullTranslations.gettext of <gettext.NullTranslations object at 0x10f6cea50>>

D: 16:23:35,094 i18n.setup_gettext:196: pgettext_attributes = <bound method NullTranslations.pgettext of <gettext.NullTranslations object at 0x10f6cea50>>
D: 16:23:35,102 webservice._network_accessible_changed:387: Network accessible requested: 1, actual: 1
D: 16:23:35,232 webservice.set_cache:409: NetworkDiskCache dir: '/Users/Nova/Library/Caches/MusicBrainz/Picard/network/' current size: 89.9 MB max size: 100 MB
D: 16:23:35,235 pluginmanager.load_plugins_from_directory:264: Looking for plugins in directory '/Users/Nova/Library/Preferences/MusicBrainz/Picard/plugins', 28 names found
D: 16:23:35,251 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='acousticbrainz' item=<bound method AcousticBrainzPlugin.process_album of <picard.plugins.acousticbrainz.AcousticBrainzPlugin object at 0x10f6d6f10>>
D: 16:23:35,251 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='acousticbrainz' item=<bound method AcousticBrainzPlugin.process_track of <picard.plugins.acousticbrainz.AcousticBrainzPlugin object at 0x10f6d6f10>>
D: 16:23:35,251 plugin.register:82: ExtensionPoint: pages register <- plugin='acousticbrainz' item=<class 'picard.plugins.acousticbrainz.AcousticBrainzOptionsPage'>
D: 16:23:35,251 pluginmanager._load_plugin:337: Loading plugin 'AcousticBrainz Tags' version 2.2.3.final0, compatible with API: 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7
D: 16:23:35,261 plugin.register:82: ExtensionPoint: cluster_actions register <- plugin='addrelease' item=<picard.plugins.addrelease.AddClusterAsRelease object at 0x10f66a9e0>
D: 16:23:35,261 plugin.register:82: ExtensionPoint: file_actions register <- plugin='addrelease' item=<picard.plugins.addrelease.AddFileAsRecording object at 0x10f66ab90>
D: 16:23:35,261 plugin.register:82: ExtensionPoint: file_actions register <- plugin='addrelease' item=<picard.plugins.addrelease.AddFileAsRelease object at 0x10f66ac20>
D: 16:23:35,262 pluginmanager._load_plugin:337: Loading plugin 'Add Cluster As Release' version 0.7.3.final0, compatible with API: 2.0
D: 16:23:35,270 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='albumartist_website' item=<bound method AlbumArtistWebsite.add_artist_website of <picard.plugins.albumartist_website.AlbumArtistWebsite object at 0x10f705290>>
D: 16:23:35,271 pluginmanager._load_plugin:337: Loading plugin 'Album Artist Website' version 1.1.0.final0, compatible with API: 2.0, 2.1, 2.2
D: 16:23:35,276 plugin.register:82: ExtensionPoint: cover_art_providers register <- plugin='amazon' item=<class 'picard.plugins.amazon.CoverArtProviderAmazon'>
D: 16:23:35,277 pluginmanager._load_plugin:337: Loading plugin 'Amazon cover art' version 1.1.0.final0, compatible with API: 2.2, 2.3, 2.4, 2.5, 2.6, 2.7
D: 16:23:35,280 plugin.register:82: ExtensionPoint: album_actions register <- plugin='collect_artists' item=<picard.plugins.collect_artists.CollectArtists object at 0x10f66b130>
D: 16:23:35,280 pluginmanager._load_plugin:337: Loading plugin 'Collect Album Artists' version 0.1.0.final0, compatible with API: 2.1, 2.2
D: 16:23:35,282 plugin.register:82: ExtensionPoint: function_registry register <- plugin='decade' item=('decade', FunctionRegistryItem(<function script_decade at 0x10f6fc2c0>, True, Bound(lower=1, upper=2), None))

D: 16:23:35,282 pluginmanager._load_plugin:337: Loading plugin 'Decade function' version 1.0.0.final0, compatible with API: 2.0, 2.1, 2.2, 2.3
D: 16:23:35,304 plugin.register:82: ExtensionPoint: cover_art_providers register <- plugin='deezerart' item=<class 'picard.plugins.deezerart.Provider'>
D: 16:23:35,305 plugin.register:82: ExtensionPoint: pages register <- plugin='deezerart' item=<class 'picard.plugins.deezerart.OptionsPage'>
D: 16:23:35,305 pluginmanager._load_plugin:337: Loading plugin 'Deezer cover art' version 1.2.0.final0, compatible with API: 2.5
D: 16:23:35,317 plugin.register:82: ExtensionPoint: cover_art_providers register <- plugin='fanarttv' item=<class 'picard.plugins.fanarttv.CoverArtProviderFanartTv'>
D: 16:23:35,317 plugin.register:82: ExtensionPoint: pages register <- plugin='fanarttv' item=<class 'picard.plugins.fanarttv.FanartTvOptionsPage'>
D: 16:23:35,318 pluginmanager._load_plugin:337: Loading plugin 'fanart.tv cover art' version 1.6.3.final0, compatible with API: 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6
D: 16:23:35,324 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='featartistsintitles' item=<function move_album_featartists at 0x10f6fe020>
D: 16:23:35,325 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='featartistsintitles' item=<function move_track_featartists at 0x10f6fe660>
D: 16:23:35,325 pluginmanager._load_plugin:337: Loading plugin 'Feat. Artists in Titles' version 0.5.0.final0, compatible with API: 2.0
D: 16:23:35,337 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='genre_mapper' item=<function track_genre_mapper at 0x10f6fe8e0>
D: 16:23:35,338 plugin.register:82: ExtensionPoint: pages register <- plugin='genre_mapper' item=<class 'picard.plugins.genre_mapper.GenreMapperOptionsPage'>
D: 16:23:35,338 //Users/Nova/Library/Preferences/MusicBrainz/Picard/plugins/genre_mapper.zip/genre_mapper.refresh:71: Genre Mapper: Refreshing the genre replacement maps processing pairs using 'Simple' translation.
D: 16:23:35,339 //Users/Nova/Library/Preferences/MusicBrainz/Picard/plugins/genre_mapper.zip/genre_mapper.refresh:101: Genre Mapper: No genre replacement maps defined.
D: 16:23:35,340 pluginmanager._load_plugin:337: Loading plugin 'Genre Mapper' version 0.5.0.final0, compatible with API: 2.0, 2.1, 2.2, 2.3, 2.6, 2.7, 2.8, 2.9
D: 16:23:35,342 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='hyphen_unicode' item=<function main at 0x10f6ff1a0>
D: 16:23:35,342 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='hyphen_unicode' item=<function main at 0x10f6ff1a0>
D: 16:23:35,342 pluginmanager._load_plugin:337: Loading plugin 'Hyphen unicode' version 1.0.1.final0, compatible with API: 2.0
D: 16:23:35,345 plugin.register:82: ExtensionPoint: function_registry register <- plugin='keep' item=('keep', FunctionRegistryItem(<function keep at 0x10f6ff2e0>, True, Bound(lower=0, upper=None), None))
D: 16:23:35,346 pluginmanager._load_plugin:337: Loading plugin 'Keep tags' version 1.2.1.final0, compatible with API: 2.0

D: 16:23:35,361 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='lastfm' item=<function process_track at 0x10f6ffd80>
D: 16:23:35,361 plugin.register:82: ExtensionPoint: pages register <- plugin='lastfm' item=<class 'picard.plugins.lastfm.LastfmOptionsPage'>
D: 16:23:35,361 pluginmanager._load_plugin:337: Loading plugin 'Last.fm' version 0.10.1.final0, compatible with API: 2.0
D: 16:23:35,366 plugin.register:82: ExtensionPoint: track_actions register <- plugin='loadasnat' item=<picard.plugins.loadasnat.LoadAsNat object at 0x10f738820>
D: 16:23:35,366 pluginmanager._load_plugin:337: Loading plugin 'Load as non-album track' version 0.4.0.final0, compatible with API: 2.0, 2.1, 2.2
D: 16:23:35,375 plugin.register:82: ExtensionPoint: file_actions register <- plugin='moodbars' item=<picard.plugins.moodbars.MoodBar object at 0x10f738f70>
D: 16:23:35,376 plugin.register:82: ExtensionPoint: pages register <- plugin='moodbars' item=<class 'picard.plugins.moodbars.MoodbarOptionsPage'>
D: 16:23:35,377 pluginmanager._load_plugin:337: Loading plugin 'Moodbars' version 2.3.3.final0, compatible with API: 2.0
D: 16:23:35,382 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='musixmatch' item=<function process_track at 0x10f748ae0>
D: 16:23:35,382 plugin.register:82: ExtensionPoint: pages register <- plugin='musixmatch' item=<class 'picard.plugins.musixmatch.MusixmatchOptionsPage'>
D: 16:23:35,383 pluginmanager._load_plugin:337: Loading plugin 'Musixmatch Lyrics' version 1.1.1.final0, compatible with API: 2.0
D: 16:23:35,386 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='non_ascii_equivalents' item=<function main at 0x10f749300>
D: 16:23:35,387 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='non_ascii_equivalents' item=<function main at 0x10f749300>
D: 16:23:35,387 pluginmanager._load_plugin:337: Loading plugin 'Non-ASCII Equivalents' version 0.4.0.final0, compatible with API: 2.0
D: 16:23:35,390 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='replace_forbidden_symbols' item=<function main at 0x10f7496c0>
D: 16:23:35,390 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='replace_forbidden_symbols' item=<function main at 0x10f7496c0>
D: 16:23:35,391 plugin.register:82: ExtensionPoint: function_registry register <- plugin='replace_forbidden_symbols' item=('replace_forbidden', FunctionRegistryItem(<function script_replace_forbidden at 0x10f749620>, True, Bound(lower=1, upper=1), None))
D: 16:23:35,391 pluginmanager._load_plugin:337: Loading plugin 'Replace Forbidden Symbols' version 0.3.0.final0, compatible with API: 2.0, 2.2
D: 16:23:35,426 plugin.register:82: ExtensionPoint: cluster_actions register <- plugin='search_engine_lookup' item=<picard.plugins.search_engine_lookup.SearchEngineLookupTest object at 0x10f739900>
D: 16:23:35,426 plugin.register:82: ExtensionPoint: pages register <- plugin='search_engine_lookup' item=<class 'picard.plugins.search_engine_lookup.SearchEngineLookupOptionsPage'>

D: 16:23:35,427 plugin.register:82: ExtensionPoint: album_actions register <- plugin='search_engine_lookup' item=<picard.plugins.search_engine_lookup.AlbumCoverArtLookup object at 0x10f739990>
D: 16:23:35,427 plugin.register:82: ExtensionPoint: track_actions register <- plugin='search_engine_lookup' item=<picard.plugins.search_engine_lookup.TrackCoverArtLookup object at 0x10f739a20>
D: 16:23:35,427 pluginmanager._load_plugin:337: Loading plugin 'Search Engine Lookup' version 2.1.0.final0, compatible with API: 2.0, 2.1, 2.2, 2.3
D: 16:23:35,432 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='smart_title_case' item=<function title_case at 0x10f74b100>
D: 16:23:35,432 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='smart_title_case' item=<function title_case at 0x10f74b100>
D: 16:23:35,432 pluginmanager._load_plugin:337: Loading plugin 'Smart Title Case' version 0.4.2.final0, compatible with API: 2.0
D: 16:23:35,434 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='sort_multivalue_tags' item=<function sort_multivalue_tags at 0x10f74b240>
D: 16:23:35,434 pluginmanager._load_plugin:337: Loading plugin 'Sort Multi-Value Tags' version 1.0.0.final0, compatible with API: 2.0
D: 16:23:35,436 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='soundtrack' item=<function soundtrack at 0x10f74b2e0>
D: 16:23:35,436 pluginmanager._load_plugin:337: Loading plugin 'Soundtrack' version 0.2.0.final0, compatible with API: 2.0
D: 16:23:35,440 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='standardise_feat' item=<function standardise_track_artist at 0x10f74b4c0>
D: 16:23:35,440 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='standardise_feat' item=<function standardise_album_artist at 0x10f74b560>
D: 16:23:35,441 pluginmanager._load_plugin:337: Loading plugin 'Standardise Feat.' version 0.3.0.final0, compatible with API: 2.0, 2.1, 2.2, 2.3
D: 16:23:35,452 plugin.register:82: ExtensionPoint: pages register <- plugin='submit_folksonomy_tags' item=<class 'picard.plugins.submit_folksonomy_tags.TagSubmitPlugin_OptionsPage'>
D: 16:23:35,453 plugin.register:82: ExtensionPoint: album_actions register <- plugin='submit_folksonomy_tags' item=<picard.plugins.submit_folksonomy_tags.SubmitTrackTagsMenuAction object at 0x10f73aa70>
D: 16:23:35,453 plugin.register:82: ExtensionPoint: track_actions register <- plugin='submit_folksonomy_tags' item=<picard.plugins.submit_folksonomy_tags.SubmitTrackTagsMenuAction object at 0x10f73ab00>
D: 16:23:35,453 plugin.register:82: ExtensionPoint: album_actions register <- plugin='submit_folksonomy_tags' item=<picard.plugins.submit_folksonomy_tags.SubmitReleaseTagsMenuAction object at 0x10f73ab90>

D: 16:23:35,453 plugin.register:82: ExtensionPoint: track_actions register <- plugin='submit_folksonomy_tags' item=<picard.plugins.submit_folksonomy_tags.SubmitReleaseTagsMenuAction object at 0x10f73ac20>
D: 16:23:35,453 plugin.register:82: ExtensionPoint: album_actions register <- plugin='submit_folksonomy_tags' item=<picard.plugins.submit_folksonomy_tags.SubmitRGTagsMenuAction object at 0x10f73acb0>
D: 16:23:35,453 plugin.register:82: ExtensionPoint: track_actions register <- plugin='submit_folksonomy_tags' item=<picard.plugins.submit_folksonomy_tags.SubmitRGTagsMenuAction object at 0x10f73ad40>
D: 16:23:35,453 plugin.register:82: ExtensionPoint: album_actions register <- plugin='submit_folksonomy_tags' item=<picard.plugins.submit_folksonomy_tags.SubmitRATagsMenuAction object at 0x10f73add0>
D: 16:23:35,453 plugin.register:82: ExtensionPoint: track_actions register <- plugin='submit_folksonomy_tags' item=<picard.plugins.submit_folksonomy_tags.SubmitRATagsMenuAction object at 0x10f73ae60>
D: 16:23:35,454 pluginmanager._load_plugin:337: Loading plugin 'Submit Folksonomy Tags' version 0.3.0.final0, compatible with API: 2.2, 2.9
D: 16:23:35,460 plugin.register:82: ExtensionPoint: album_actions register <- plugin='submit_isrc' item=<picard.plugins.submit_isrc.SubmitAlbumISRCs object at 0x10f73b2e0>
D: 16:23:35,461 pluginmanager._load_plugin:337: Loading plugin 'Submit ISRC' version 1.1.0.final0, compatible with API: 2.0, 2.1, 2.2, 2.3, 2.6, 2.9
D: 16:23:35,462 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='titlecase' item=<function title_case at 0x111654d60>
D: 16:23:35,462 plugin.register:82: ExtensionPoint: album_metadata_processors register <- plugin='titlecase' item=<function title_case at 0x111654d60>
D: 16:23:35,462 pluginmanager._load_plugin:337: Loading plugin 'Title Case' version 1.0.2.final0, compatible with API: 2.0
D: 16:23:35,466 plugin.register:82: ExtensionPoint: cluster_actions register <- plugin='tracks2clipboard' item=<picard.plugins.tracks2clipboard.CopyClusterToClipboard object at 0x10f73b5b0>
D: 16:23:35,466 pluginmanager._load_plugin:337: Loading plugin 'Copy Cluster to Clipboard' version 1.0.0.final0, compatible with API: 2.0
D: 16:23:35,477 plugin.register:82: ExtensionPoint: track_metadata_processors register <- plugin='wikidata' item=<bound method Wikidata.process_track of <picard.plugins.wikidata.Wikidata object at 0x11163c510>>
D: 16:23:35,478 plugin.register:82: ExtensionPoint: pages register <- plugin='wikidata' item=<class 'picard.plugins.wikidata.WikidataOptionsPage'>

D: 16:23:35,478 pluginmanager._load_plugin:337: Loading plugin 'Wikidata Genre' version 1.4.5.final0, compatible with API: 2.0, 2.1, 2.2

D: 16:23:35,496 ui/playertoolbar.__init__:91: Internal player: QtMultimedia available, initializing QMediaPlayer
D: 16:23:35,563 ui/playertoolbar.__init__:98: Internal player: available, QMediaPlayer set up
D: 16:23:36,287 util/periodictouch.enable_timer:47: Setup timer for touching files every 14400 seconds
D: 16:23:36,290 tagger.main:1576: Looking for Qt locale en_US in /Applications/MusicBrainz Picard.app/Contents/MacOS/PyQt5/Qt5/translations

D: 16:23:36,384 ui/mainwindow.auto_update_check:1786: Skipping startup check for program updates. Today: 2024-01-26, Last check: 2024-01-21 (Check interval: 7 days), Update level: 1 (beta)
D: 16:23:36,469 config.event:261: Config file update requested on thread 4536618496
D: 16:23:36,718 webservice/ratecontrol.get_delay_to_next_request:122: ('musicbrainz.org', 443): First request
D: 16:23:36,719 webservice/ratecontrol.increment_requests:147: ('musicbrainz.org', 443): Incrementing requests to: 1
D: 16:23:38,259 webservice/ratecontrol.get_delay_to_next_request:122: ('picard.musicbrainz.org', 443): First request
D: 16:23:38,259 webservice/ratecontrol.increment_requests:147: ('picard.musicbrainz.org', 443): Incrementing requests to: 1
D: 16:23:39,256 webservice/ratecontrol.decrement_requests:155: ('picard.musicbrainz.org', 443): Decrementing requests to: 0
D: 16:23:39,257 webservice._handle_reply:558: Received reply for https://picard.musicbrainz.org/api/v2/plugins/ -> HTTP 200 (OK)
D: 16:23:39,259 webservice._handle_reply:571: Response received: {'plugins': {'abbreviate_artistsort': {'api_versions': ['1.0', '2.0'], 'author': 'Sophist', 'description': ' 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%. ', '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': ' 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. ', 'files': {'__init__.py': 'c731ef9137dfaeea18950c9f95bf9453',

'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': " Add's the following tags: \n
\n
Key (in ID3v2.3 format)
\n
Beats Per Minute (BPM)
\n
\n from the AcousticBrainz database.

\nThis plugin is deprecated, please consider using the AcousticBrainz Tags\nplugin instead. ", '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'}, 'additional_artists_details': {'api_versions': ['2.0', '2.1', '2.2', '2.7', '2.8'], 'author': 'Bob Swift (rdswift)', 'description': ' 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

\nPlease see the user\nguide on GitHub for more information. ', 'files': {'__init__.py': 'c045ba6f5d5e5da694627dc8bb17dee9', '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': '0.3'}, 'additional_artists_variables': {'api_versions': ['2.0', '2.1', '2.2', '2.7', '2.9', '2.10'], 'author': 'Bob Swift (rdswift)', 'description': ' 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

\nPlease see the user guide on GitHub for more information. ', 'files': {'additional_artists_variables.py': 'cf708f446e653d56202a8bf25638e4d8'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Additional Artists Variables', 'version': '0.9'}, 'addrelease': {'api_versions': ['2.0'], 'author': 'Frederik "Freso" S. Olesen, Lukáš Lalinský, Philip Jägenstedt', 'description': ' 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. ', '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': " Add's the album artist(s)

Official Homepage(s)\n(if they are defined in the MusicBrainz database). ", 'files': {'albumartist_website.py': 'd65fd19fa2f45c7bfe687c981d9832c4'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Album Artist Website', 'version': '1.1'}, 'albumartistextension': {'api_versions': ['2.0'], 'author': 'Bob Swift (rdswift)', 'description': ' 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: \n<ul>\n<li>_aaeStdAlbumArtists = The standardized version of the album artists.\n<li>_aaeCredAlbumArtists = The credited version of the album artists.\n<li>_aaeSortAlbumArtists = The sorted version of the album artists.\n<li>_aaeStdPrimaryAlbumArtist = The standardized version of the first\n   (primary) album artist.\n
_aaeCredPrimaryAlbumArtist = The credited version of the first (primary)\n album artist.\n
_aaeSortPrimaryAlbumArtist = The sorted version of the first (primary)\n album artist.\n
_aaeAlbumArtistCount = The number of artists comprising the album artist.\n
\n 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. ', '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': ' Use cover art from Amazon. ', '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': ' Calculate BPM for selected files and albums. Linux only version with dependancy on Aubio and Numpy ', 'files': {'__init__.py': '86fee79b8805d4862fa41c9ebb987cc9', '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': '1.5.1'}, 'classical_extras': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.5', '2.6', '2.7'], 'author': 'Mark Evens', 'description': ' 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\nuser interface in Picard options->plugins. This interface provides separate sections\nto enhance artist/performer tags, works and parts, genres and also allows for a generalised\n"tag mapping" (simple scripting).\nWhile it is designed to cater for the complexities of classical music tagging,\nit may also be useful for other music which has more than just basic

song/artist/album data.\n

\nThe options screen provides five tabs for users to control the tags produced:\n

\n1. Artists: Options as to whether artist tags will contain standard MB names, aliases or as-credited names.\nAbility to include and annotate names for specialist roles (chorus master, arranger, lyricist etc.).\nAbility to read lyrics tags on the file which has been loaded and assign them to track and album levels if required.\n(Note: Picard will not normally process incoming file tags).\n

\n2. Works and parts: The plugin will build a hierarchy of works and parts (e.g. Work -> Part -> Movement or\nOpera -> Act -> Number) based on the works in MusicBrainz\'s database. These can then be displayed in tags in a variety\nof ways according to user preferences. Furthermore partial recordings, medleys, arrangements and collections of works\nare all handled according to user choices. There is a processing overhead for this at present because MusicBrainz limits\nlook-ups to one per second.\n

\n3. Genres etc.: Options are available to customise the source and display of information relating to genres,\ninstruments, keys, work dates and periods. Additional capabilities are provided for users of Muso (or others who\nprovide the relevant XML files) to use pre-existing databases of classical genres, classical composers and classical\nperiods.\n

\n4. Tag mapping: in some ways, this is a simple substitute for some of Picard\'s scripting capability. The main advantage\n is that the plugin will remember what tag mapping you use for each release (or even track).\n

\n5. Advanced: Various options to control the detailed processing of the above.\n

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

\nSee the readme file \non 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': ' Moves disc numbers and subtitles from the separate tags to album titles. ', 'files': {'classicdiscnumber.py': 'ec4abd529133d83ea0727db3d0e99699'}, 'name': 'Classic Disc Numbers', 'version': '0.2'}, 'collect_artists': {'api_versions': ['2.1', '2.2'], 'author': 'johbi', 'description': ' Adds a context menu shortcut to collect all track artists from a release and format them as the releases album artist. ', 'files': {'collect_artists.py': 'adef9c3fcdb5bcb4085bb493c45f7575'}, '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': " 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. ", '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':

' Uses Critiquebrainz for comment as review or rating. \n WARNING: Experimental plugin. All guarantees voided by use. \n Example: Taylor Swift\nRelease: Midnights\n https://musicbrainz.org/release/e348fdd6-f73b-47fe-94c4-670bfee26a39 ,\n https://critiquebrainz.org/release-group/0dcc84fb-c592-46e9-ba92-a52bb44dd553 \n Recording:\n https://musicbrainz.org/recording/93113326-93e9-409c-a3d6-5ec91864ba30 ,\n https://critiquebrainz.org/recording/93113326-93e9-409c-a3d6-5ec91864ba30 ', 'files': {'critiquebrainz.py': 'c9c4ab9191a0102f4f9a6d5f7cc50041'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Critiquebrainz Review Comment', 'version': '1.0.1'}, 'cuesheet': {'api_versions': ['2.0'], 'author': 'Lukáš Lalinský, Sambhav Kothari', 'description': ' Generate cuesheet (.cue file) from an album. ', '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': ' 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". ', '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': ' This plugin helps you quickly convert mis-encoded cyrillic Windows-1251 tags\nto proper UTF-8 encoded strings. If your track/album names look something like\n"Àëêñà â ñò›àíå ÷óääñ", run this plugin from the context menu\nbefore running the "Lookup" or "Scan" tools ', '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': ' This plugin helps you quickly convert mis-encoded Greek Windows-1253 tags\nto proper UTF-8 encoded strings. If your track/album names look something like\n"Àëêñà â ñò›àíå ÷óääñ", run this plugin from the context menu\nbefore running the "Lookup" or "Scan" tools ', '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': ' Fetch cover arts from Deezer ', 'files': {'__init__.py': 'da466949d7971ddd0f50d2302523e60e', 'deezer/__init__.py': 'f35ef270482c4cde5341b98c74612b9f', 'deezer/client.py': '45697b0b1424ff4afd8ae1df04cc2c6a', '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'}, 'discnumber': {'api_versions': ['0.9.0', '0.10', '0.15', '2.0'], 'author': 'Lukas Lalinsky', 'description': ' Moves disc numbers and subtitles from album titles to separate tags. For example:
\n"Aerial (disc 1: A Sea of Honey)" \n
\n album = "Aerial"
\n discnumber = "1"
\n discsubtitle = "A Sea of Honey"
\n
', 'files': {'discnumber.py': 'e5e63a013534bdb2be6c8b3899d3f336'}, 'name': 'Disc Numbers', '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': ' Use cover art from fanart.tv.

To use this plugin you have to register a personal API key on fanart.tv. ', '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': ' Removes feat. artists from track titles. Substitution is case insensitive. ', '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': ' Move "feat." from artist names to album and track titles. Match is case insensitive. ', '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': ' Fix the track numbers in a cluster by either using the track titles (1) or sequential order (2). \n<ol>\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 The track numbers will be set based on the sequential order they appear within the cluster.
\n
\n\n How to use: \n
\n Cluster a group of files
\n Right click on the cluster
\n \n   Then click one:\n   \n   Plugins => Fix track numbers using titles
\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': " 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. ", 'files': {'__init__.py': '542929a88d9f87040e4b634bb1eb40a7', 'docs/HISTORY.md': '23336f12177011aa18533aafeaf224fd', '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': '0.8.1'}, 'genre_mapper': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.6', '2.7', '2.8', '2.9'], 'author': 'Bob Swift', 'description': ' 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. ', 'files': {'__init__.py': '4876a1980fe9784d5732d5835d492a56', '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': '0.5'}, 'haikuattrs': {'api_versions': ['2.2', '2.3', '2.4', '2.5', '2.6'], 'author': 'Philipp Wolfer', 'description': ' Save and load metadata to/from Haiku BFS attributes. ', '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'}, 'hyphen_unicode': {'api_versions': ['0.9', '0.10', '0.11', '0.15', '2.0'], 'author': 'Alan Swanson <revier@improbability.net>', 'description': ' Replaces unicode character HYPHEN (U+2010) [0xE2 0x80\n0x90] with typographically identical HYPHEN-MINUS (U+002D) [0x2D] for fonts\nthat do not support HYPHEN and to prevent visually duplicate filenames\ndifferentiated only by their hyphens. \n Unicode duplicated hyphen from ASCII as an unambiguous way to designate a\nhyphen from a minus whilst still being typographically indentical. Since\ntext processing on music tags is rare so choice is purely pedantic esepcially\nas keyboards only have HYPHEN-MINUS. \n Replaces character on "album", "title", "artist", "artists", "artistsort",\n"albumartist", "albumartists" and "albumartistsort" tags. ', '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': ' Adds a multi-valued tag (~instruments) containing all the instruments (including vocals), \n for use in scripts. ', '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': ' 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>. \n 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. ', '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'], 'author': 'Bob Swift', 'description': " Adds functions to convert between 'standard', 'camelot', 'open key' and 'traktor' key formats. ", 'files': {'key_wheel_converter.py': 'fdc3cf9eb060a0590b561d2bdb47b028'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.txt', 'name': 'Key Wheel Converter', 'version': '1.1'}, 'lastfm': {'api_versions': ['2.0'], 'author': 'Lukáš Lalinský, Philipp Wolfer', 'description': ' Use tags from Last.fm as genre. ', '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': ' Allows loading selected tracks as non-album tracks. Useful for tagging single tracks where you do not care about the album. ', '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'}, 'mod': {'api_versions': ['2.8'], 'author': 'Philipp Wolfer', 'description': ' 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. ', 'files': {'__init__.py': 'da669511333874800332 00d856f2886e'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'MOD files', 'version': '0.1'}, 'moodbars': {'api_versions': ['2.0'], 'author': 'Len Joubert, Sambhav Kothari', 'description': ' Calculate Moodbars for selected files and albums.

\nAccording to WikiPedia\na "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 <a href="http://userbase.kde.org/Amarok/Manual/Various/Moodbar">Amarok Moodbar page<a> for details). ', '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': ' Fetch first 30% of lyrics from Musixmatch ', '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': ' Do not store specific release information in releases of unknown origin. ', '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>', 'description': ' 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. \n Currently replaces characters on "album", "artist", and "title" tags. ', 'files': {'non_ascii_equivalents.py': '0e91d256974b7acc181b86b8779eba3d'}, 'license': 'GPL-3.0-or-later', 'license_url': 'https://gnu.org/licenses/gpl.html', 'name': 'Non-ASCII Equivalents', 'version': '0.4'}, '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': ' 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. ', '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': ' Create a paper CD case from an album or cluster using http://papercdcase.com ', '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'}, 'persistent_variables': {'api_versions': ['2.0', '2.1', '2.2', '2.3', '2.4', '2.6', '2.7'], 'author': 'Bob Swift (rdswift)', 'description': ' \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 \n \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\nalbum is refreshed. The information is cleared when an album is removed. Session variables persist across all\nalbums and tracks, and are cleared when Picard is shut down or restarted.\n \n \nThis plugin adds eight new scripting functions to allow management of persistent script variables:\n
\n
$set_a(name,value) : Sets the album persistent variable name to value.
\n
$unset_a(name) : Unsets the album persistent variable name.
\n
$get_a(name) : Gets the album persistent variable name.
\n
$clear_a() : Clears all album persistent variables.
\n
$set_s(name,value) : Sets the session persistent variable name to value.
\n
$unset_s(name) : Unsets the session persistent variable name.
\n
$get_s(name) : Gets the session persistent variable name.
\n
$clear_s() : Clears all session persistent variables.
\n
\n \n \nPlease see the user guide on GitHub for more information.\n ', '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': ' 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. ', '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'}, 'release_type': {'api_versions': ['0.9.0', '0.10', '0.15', '2.0'], 'author': 'Elliot Chance', 'description': ' Appends information to EPs and Singles ', 'files': {'release_type.py': 'c66f0989f9ebcc94bc3ed183806e8b91'}, 'name': 'Release Type', 'version': '1.4'}, 'releasetag_aggregations': {'api_versions': ['2.5', '2.6'], 'author': 'Philipp Wolfer', 'description': ' Add functions to aggregate tags on a release:
$album_all(name)
\u200e$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(na
me, separator=;)
The functions work only in file naming scripts and the files should
either be part of a release or cluster! ', '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': ' Remove all perfectly matched albums from the selection. ',
'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': ' 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, ... \n 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. ', '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':
' 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%)) ', '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': ' 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.\nThis plugin depends on the ReplayGain utility <a
href="https://github.com/complexlogic/rsgain">rsgain. Users\nare required to install rsgain
and set its path in the plugin settings before use. \n<h4>Usage</h4>\n Select one or
more tracks or albums, then right click and select Plugin->Calculate ReplayGain. The
plugin\nwill calculate ReplayGain information for the selected items and display the results in
the metadata\nwindow. Click the save button to write the tags to file. \n The following file
formats are supported: \n
\n
MP3 (.mp3)
\n
FLAC (.flac)
\n<li>Ogg (.ogg,
.oga, spx)
\n
Opus (.opus)
\n
MPEG-4 Audio (.m4a, .mp4)
\n<li>Wavpack
(.wv)
\n
Monkey\'s Audio (.ape)
\n
WMA (.wma)
\n<li>MP2
(.mp2)
\n
WAV (.wav)
\n
AIFF (.aiff)
\n
TAK (.tak)
\n<li>MusePack

(Stream Version 8 only) (.mpc)
\n
\n This plugin is based on the original ReplayGain plugin by Philipp Wolfer and Sophist. ', 'files': {'__init__.py': '69cf4b2738c3522c176b67eb789a0bbf', 'ui_options_replaygain2.py': '4bae8b24cbc5057353ee95bf08835209', 'ui_options_replaygain2.ui': 'dc399977085c4cf5bc9001f87ad0517f'}, 'license': 'GPL-2.0', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'ReplayGain 2.0', 'version': '1.6'}, 'save_and_rewrite_header': {'api_versions': ['0.9.0', '0.10', '0.15', '2.0'], 'author': 'Nicolas Cenerario', 'description': ' This plugin adds a context menu action to save files and rewrite their header. ', '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': " This plugin provides a new script function <code>$logline()</code> to write entries\nto Picard's system log. By default, the log level is set at <code>Info</code>, but\nany level can be used by providing the level as an optional second\nparameter to the function. \n The function is used as: \n <code>$logline(text[,level])</code> \n 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: \n
\n
E (Error)
\n
W (Warning)
\n
I (Info)
\n
D (Debug)
\n
\n If an unknown level is entered, the function will use the default <code>Info</code>\nlevel. ", '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': ' Adds a right click option on a cluster to look up album information using a search engine in a browser window. \n Adds a right click option on an album or track to look up cover art for the selected album or title. ', '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': ' 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. ', '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': ' This plugin sorts multi-value tags e.g. Performers alphabetically.

\nNote: Some multi-value tags are excluded for the following reasons: \n
\n
Sequence is important e.g. Artists
\n<li>The sequence of one tag is linked to the sequence of another e.g. Label and

Catalogue number.
\n
', 'files': {'sort_multivalue_tags.py': '4f25a8402f5ce6a44427bd4df94af766'}, 'license': 'GPL-2.0-or-later', 'license_url': 'https://www.gnu.org/licenses/gpl-2.0.html', 'name': 'Sort Multi-Value Tags', 'version': '1.0'}, 'soundtrack': {'api_versions': ['1.0', '2.0'], 'author': 'Samir Benmendil', 'description': ' Sets the albumartist to "Soundtrack" if releasetype is a soundtrack. ', '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': ' Standardises "featuring" join phrases for artists to "feat." ', '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': ' Splits multi-instrument performer tags into single\ninstruments and combines names so e.g. (from 10cc by 10cc track 1): \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 becomes: \n<pre>\nPerformer [acoustic guitar]: Graham Gouldman; Lol Creme\nPerformer [bass]: Graham Gouldman\nPerformer [dobro]: Graham Gouldman\nPerformer [electric guitar]: Eric Stewart; Graham Gouldman; Lol Creme\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 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. ', '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': ' A plugin allowing submission of specific tags on tracks you own (defaults to genre and mood) as folksonomy tags on MusicBrainz. Supports submitting to recording, release, release group and release artist entities. \n 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 Plugins and depending on what you want to submit, choose the option you want. \n Uses code from rdswift\'s "Submit ISRC" plugin (specifically, the handling of the network response) ', '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': ' \nAdds a right click option on an album to submit the ISRCs to the MusicBrainz server\nspecified in the Options.\n \n \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 \n \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 \n \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 \n \nWhen ISRCs have been submitted, a notice will be displayed showing whether or not\nthe submission was successful.\n ', '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': ' Load genre, date and vocalist tags for latin dance music\nfrom tango.info. ', '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': ' Use cover art from TheAudioDB. ', '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': ' Capitalize First Character In Every Word Of A Title ', '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': " Exports a cluster's tracks to the clipboard, so it can be copied into the tracklist field on MusicBrainz ", 'files': {'tracks2clipboard.py': 'c9efa9499819980dcfc5f565ce6a2108'}, 'name': 'Copy Cluster to Clipboard', 'version': '1.0'}, 'viewvariables': {'api_versions': ['2.0'], 'author': 'Sophist', 'description': ' Display a dialog box listing the metadata variables for the track / file.

\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. ', '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': ' Query wikidata to get genre tags ', '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': ' Set work and movement based on work relationships. \n 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. \n 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. \n 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. ', '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: 16:23:39,263 webservice/ratecontrol._out_of_backoff:231: ('picard.musicbrainz.org', 443): oobackoff; delay: 1000ms -> 1000ms; slow start; window size 1.000 -> 2.000
D: 16:23:39,344 webservice/ratecontrol.decrement_requests:155: ('musicbrainz.org', 443): Decrementing requests to: 0
D: 16:23:39,344 webservice._handle_reply:558: Received reply for https://musicbrainz.org/ws/2/collection -> HTTP 200 (OK)
D: 16:23:39,344 webservice._handle_reply:571: Response received: {'collection-offset': 0, 'collection-count': 0, 'collections': []}
D: 16:23:39,344 collection.request_finished:142: User collections: []
D: 16:23:39,344 webservice/ratecontrol._out_of_backoff:231: ('musicbrainz.org', 443): oobackoff; delay: 1000ms -> 1000ms; slow start; window size 1.000 -> 2.000