nexus3-cli Documentation

Thiago Figueiró

Feb 16, 2020

Contents:

CHAPTER 1

Command-line Interface

Logging level can be configured by setting an environment variable named `LOG_LEVEL`. Valid values are: `DEBUG`, `INFO`, `WARNING` (default), `ERROR`, `CRITICAL`.

CHAPTER 2

API

2.1 nexuscli

2.1.1 nexuscli package

Subpackages

nexuscli.api package

Subpackages

nexuscli.api.cleanup_policy package

Submodules

nexuscli.api.cleanup_policy.collection module

class nexuscli.api.cleanup_policy.collection.**CleanupPolicyCollection**(*client=None*)
Bases: object

A class to manage Nexus 3 Cleanup Policies.

Parameters **client** (nexuscli.nexus_client.NexusClient) – the client instance that will be used to perform operations against the Nexus 3 service. You must provide this at instantiation or set it before calling any methods that require connectivity to Nexus.

GROOVY_SCRIPT_NAME = 'nexus3-cli-cleanup-policy'
Groovy script used by this class

create_or_update(*cleanup_policy*)
Creates the given Cleanup Policy in the Nexus repository. If a policy with the same name already exists, it will be updated.

Parameters **cleanup_policy** (CleanupPolicy) – the policy to create or update.

Raises ***exception.NexusClientCreateCleanupPolicyError*** – when the Nexus API returns an error or unexpected result.

get_by_name(*name*)

Get a Nexus 3 cleanup policy by its name.

Parameters **name** (*str*) – name of the wanted policy

Returns the requested object

Return type *CleanupPolicy*

Raises ***exception.NexusClientInvalidRepository*** – when a repository with the given name isn't found.

list()

Return all cleanup policies.

Returns every policy as a list of *CleanupPolicy* instances.

Return type list[*CleanupPolicy*]

nexuscli.api.cleanup_policy.model module

class nexuscli.api.cleanup_policy.model.**CleanupPolicy**(*client*, ***kwargs*)

Bases: object

Represents a Nexus Cleanup Policy.

Parameters

- **client** (nexuscli.nexus_client.NexusClient) – the client instance that will be used to perform operations against the Nexus 3 service. You must provide this at instantiation or set it before calling any methods that require connectivity to Nexus.
- **name** (*str*) – name of the new policy.
- **format** (*str*) – 'all' or the name of the repository format this policy applies to.
- **mode** (*str*) – 'delete'
- **criteria** (*dict*) – the deletion criteria for the policy. Supports one or more of the following attributes:
  - `lastDownloaded` (int): days since artefact last downloaded;
  - `lastBlobUpdated` (int): days since last update to artefact;

configuration

Nexus 3 Cleanup Policy representation as a python dict. The dict returned by this property can be converted to JSON for use with the `nexus3-cli-cleanup-policy` groovy script created by the *CleanupPolicyCollection* methods.

Example structure and attributes common to all repositories:

```
>>> cleanup_policy = {
>>>     'name': 'my-policy',
>>>     'format': 'bower',
>>>     'mode': 'delete',
>>>     'criteria': {
>>>         'lastDownloaded': 172800,
```

(continues on next page)

(continued from previous page)

```
>>>         'lastBlobUpdated': 86400
>>>     }
>>> }
```

Depending on the repository type and format (recipe), other attributes will be present.

Returns cleanup policy as a dict

Return type dict

nexuscli.api.cleanup_policy.validations module

nexuscli.api.cleanup_policy.validations.**policy_criteria**(*raw_policy*)
Ensures the policy criteria fields are valid. Will transform strings to the correct type where needed.

Parameters **raw_policy** – as returned by the *CleanupPolicy configuration*

Raises **ValueError** – when a criterion has an invalid value.

nexuscli.api.cleanup_policy.validations.**policy_name**(*raw_policy*)
Ensure the policy has a name

Raises **ValueError** – when the name attribute is missing.

Module contents

nexuscli.api.repository package

Submodules

nexuscli.api.repository.collection module

class nexuscli.api.repository.collection.**RepositoryCollection**(*client=None*)
Bases: object

A class to manage Nexus 3 repositories.

Parameters **client** (nexuscli.nexus_client.NexusClient) – the client instance that will be used to perform operations against the Nexus 3 service. You must provide this at instantiation or set it before calling any methods that require connectivity to Nexus.

create(*repository*)
Creates a Nexus repository with the given format and type.

Parameters **repository** (Repository) – the instance containing the settings for the repository to be created.

Raises ***NexusClientCreateRepositoryError*** – error creating repository.

delete(*name*)
Delete a repository.

Parameters **name** (*str*) – name of the repository to be deleted.

get_by_name(*name*)
Get a Nexus 3 repository by its name.

Parameters **name** (*str*) – name of the repository wanted

Return type *nexuscli.api.repository.model.Repository*

Raises ***exception.NexusClientInvalidRepository*** – when a repository with the given name isn't found.

get_raw_by_name(*name*)
Return the raw dict for the repository called `name`. Remember to *refresh()* to get the latest from the server.

Parameters **name** (*str*) – name of the repository wanted

Return type dict

Raises ***exception.NexusClientInvalidRepository*** – when a repository with the given name isn't found.

raw_list()
A raw representation of the Nexus repositories.

Returns for the format, see List Repositories.

Return type dict

refresh()
Refresh local list of repositories with latest from service. A raw representation of repositories can be fetched using *raw_list()*.

nexuscli.api.repository.collection.**get_repository_class**(*raw_configuration*)
Given a raw repository configuration, returns its corresponding class.

Parameters **raw_configuration** (*dict*) – configuration as returned by the SCRIPT_NAME_GET groovy script.

Returns repository class

nexuscli.api.repository.model module

class nexuscli.api.repository.model.**BowerHostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)
Bases: *nexuscli.api.repository.model.HostedRepository*

class nexuscli.api.repository.model.**BowerProxyRepository**(*name*, *remote_url=None*, *auto_block=True*, *content_max_age=1440*, *metadata_max_age=1440*, *negative_cache_enabled=True*, *negative_cache_ttl=1440*, ***kwargs*)
Bases: *nexuscli.api.repository.model.ProxyRepository*

class nexuscli.api.repository.model.**HostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)
Bases: *nexuscli.api.repository.model.Repository*

A hosted Nexus repository.

Parameters

- **name** (*str*) – name of the repository.
- **write_policy** (*str*) – one of *WRITE_POLICIES*. See Nexus documentation for details.
- **kwargs** – see *Repository*

TYPE = 'hosted'

WRITE_POLICIES = ['ALLOW', 'ALLOW_ONCE', 'DENY']
Nexus 3 repository write policies supported by this class.

configuration
As per *Repository.configuration* but specific to this repository recipe and type.

Return type str

upload_directory(*src_dir*, *dst_dir*, *recurse=True*, *flatten=False*)
Uploads all files in a directory to the specified destination directory in this repository, honouring options flatten and recurse.

Parameters

- **src_dir** – path to local directory to be uploaded
- **dst_dir** – destination directory in dst_repo
- **recurse** (*bool*) – when True, upload directory recursively.
- **flatten** – when True, the source directory tree isn't replicated on the destination.

Returns number of files uploaded

Return type int

upload_file(*src_file*, *dst_dir*, *dst_file=None*)
Uploads a singe file to the directory and file name specified.

Parameters

- **src_file** – path to the local file to be uploaded.
- **dst_dir** – directory under dst_repo to place file in.
- **dst_file** – destination file name. If not given, the basename of `src_file` name is used.

class nexuscli.api.repository.model.**MavenHostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)
Bases: *nexuscli.api.repository.model.HostedRepository*, *nexuscli.api.repository.model.MavenRepository*

A Maven hosted Nexus repository.

See *HostedRepository* and *MavenRepository*

class nexuscli.api.repository.model.**MavenProxyRepository**(*name*, *layout_policy='PERMISSIVE'*, *version_policy='RELEASE'*, ***kwargs*)
Bases: *nexuscli.api.repository.model.MavenRepository*, *nexuscli.api.repository.model.ProxyRepository*

A Maven proxy Nexus repository.

See *MavenRepository* and *ProxyRepository*

class nexuscli.api.repository.model.**MavenRepository**(*name*, *layout_policy='PERMISSIVE'*, *version_policy='RELEASE'*, ***kwargs*)

Bases: *nexuscli.api.repository.model.Repository*

A base Maven Nexus repository.

Parameters

- **name** (*str*) – name of the repository.
- **layout_policy** – one of *LAYOUT_POLICIES*. See Nexus documentation for details.
- **version_policy** – one of *VERSION_POLICIES*. See Nexus documentation for details.
- **kwargs** – see *Repository*

LAYOUT_POLICIES = ('PERMISSIVE', 'STRICT')

Maven layout policies

RECIPES = ('maven',)

VERSION_POLICIES = ('RELEASE', 'SNAPSHOT', 'MIXED')

Maven version policies

configuration

As per *Repository.configuration* but specific to this repository recipe and type.

Return type str

class nexuscli.api.repository.model.**NpmHostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)

Bases: *nexuscli.api.repository.model.HostedRepository*

class nexuscli.api.repository.model.**NpmProxyRepository**(*name*, *remote_url=None*, *auto_block=True*, *content_max_age=1440*, *metadata_max_age=1440*, *negative_cache_enabled=True*, *negative_cache_ttl=1440*, ***kwargs*)

Bases: *nexuscli.api.repository.model.ProxyRepository*

class nexuscli.api.repository.model.**NugetHostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)

Bases: *nexuscli.api.repository.model.HostedRepository*

class nexuscli.api.repository.model.**NugetProxyRepository**(*name*, *remote_url=None*, *auto_block=True*, *content_max_age=1440*, *metadata_max_age=1440*, *negative_cache_enabled=True*, *negative_cache_ttl=1440*, ***kwargs*)

Bases: *nexuscli.api.repository.model.ProxyRepository*

class nexuscli.api.repository.model.**ProxyRepository**(*name*, *remote_url=None*, *auto_block=True*, *content_max_age=1440*, *metadata_max_age=1440*, *negative_cache_enabled=True*, *negative_cache_ttl=1440*, ***kwargs*)

Bases: *nexuscli.api.repository.model.Repository*

A proxy Nexus repository.

Parameters

- **name** (*str*) – name of the repository.
- **remote_url** (*str*) – The URL of the repository being proxied, including the protocol scheme.
- **auto_block** (*bool*) – Auto-block outbound connections on the repository if remote peer is detected as unreachable/unresponsive.
- **content_max_age** (*int*) – How long (in minutes) to cache artifacts before rechecking the remote repository. Release repositories should use -1.
- **metadata_max_age** (*int*) – How long (in minutes) to cache metadata before rechecking the remote repository.
- **negative_cache_enabled** (*bool*) – Cache responses for content not present in the proxied repository
- **negative_cache_ttl** (*int*) – How long to cache the fact that a file was not found in the repository (in minutes)
- **kwargs** – see *Repository*

TYPE = 'proxy'

configuration

As per *Repository.configuration* but specific to this repository recipe and type.

Return type str

class nexuscli.api.repository.model.**PypiHostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)

Bases: *nexuscli.api.repository.model.HostedRepository*

class nexuscli.api.repository.model.**PypiProxyRepository**(*name*, *remote_url=None*, *auto_block=True*, *content_max_age=1440*, *metadata_max_age=1440*, *negative_cache_enabled=True*, *negative_cache_ttl=1440*, ***kwargs*)

Bases: *nexuscli.api.repository.model.ProxyRepository*

class nexuscli.api.repository.model.**RawHostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)

Bases: *nexuscli.api.repository.model.HostedRepository*

class nexuscli.api.repository.model.**RawProxyRepository**(*name*, *remote_url=None*, *auto_block=True*, *content_max_age=1440*, *metadata_max_age=1440*, *negative_cache_enabled=True*, *negative_cache_ttl=1440*, ***kwargs*)

Bases: *nexuscli.api.repository.model.ProxyRepository*

class nexuscli.api.repository.model.**Repository**(*name*, *nexus_client=None*, *recipe='raw'*, *blob_store_name='default'*, *strict_content_type_validation=False*, *cleanup_policy=None*)

Bases: object

A base Nexus repository.

Nexus 3 repository recipes (formats) supported by this class:

- bower
- npm
- nuget
- pypi
- raw
- rubygems

Parameters

- **name** (*str*) – name of the repository.
- **nexus_client** (nexuscli.nexus_client.NexusClient) – the *NexusClient* instance that will be used to perform operations against the Nexus 3 service. You must provide this at instantiation or set it before calling any methods that require connectivity to Nexus.
- **recipe** (*str*) – format (recipe) of the new repository. Must be one of *RECIPES*. See Nexus documentation for details.
- **blob_store_name** (*str*) – name of an existing blob store; 'default' should work on most installations.

- **strict_content_type_validation** (*bool*) – Whether to validate file extension against its content type.
- **cleanup_policy** (*str*) – name of an existing repository clean-up policy.

RECIPES = ('bower', 'npm', 'nuget', 'pypi', 'raw', 'rubygems')

TYPE = None

cleanup_policy
Groovy-formatted value for the cleanup/policy attribute.

configuration
Repository configuration represented as a python dict. The dict returned by this property can be converted to JSON for use with the `nexus3-cli-repository-create` groovy script created by the *create()* method.

Example structure and attributes common to all repositories:

```
>>> common_configuration = {
>>>     'name': 'my-repository',
>>>     'online': True,
>>>     'recipeName': 'raw',
>>>     '_state': 'present',
>>>     'attributes': {
>>>         'storage': {
>>>             'blobStoreName': 'default',
>>>         },
>>>         'cleanup': {
>>>             'policyName': None,
>>>         }
>>>     }
>>> }
```

Depending on the repository type and format (recipe), other attributes will be present.

Returns repository configuration

Return type dict

recipe_name
The Nexus 3 name for this repository's recipe (format). This is almost always the same as `name` with `maven` being the notable exception.

class nexuscli.api.repository.model.**RubygemsHostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)

Bases: *nexuscli.api.repository.model.HostedRepository*

class nexuscli.api.repository.model.**RubygemsProxyRepository**(*name*, *remote_url=None*, *auto_block=True*, *content_max_age=1440*, *metadata_max_age=1440*, *negative_cache_enabled=True*, *negative_cache_ttl=1440*, ***kwargs*)

Bases: *nexuscli.api.repository.model.ProxyRepository*

class nexuscli.api.repository.model.**YumHostedRepository**(*name*, *write_policy='ALLOW'*, ***kwargs*)

Bases: *nexuscli.api.repository.model.HostedRepository*, *nexuscli.api.repository.model.YumRepository*

A Yum hosted Nexus repository.

See *HostedRepository* and *YumRepository*

class nexuscli.api.repository.model.**YumProxyRepository**(*name*, *remote_url=None*, *auto_block=True*, *content_max_age=1440*, *metadata_max_age=1440*, *negative_cache_enabled=True*, *negative_cache_ttl=1440*, ***kwargs*)

Bases: *nexuscli.api.repository.model.ProxyRepository*, *nexuscli.api.repository.model.YumRepository*

A Yum proxy Nexus repository.

See *ProxyRepository* and *YumRepository*

class nexuscli.api.repository.model.**YumRepository**(*name*, *depth=1*, ***kwargs*)

Bases: *nexuscli.api.repository.model.Repository*

A Yum base Nexus repository.

Parameters

- **name** (*str*) – name of the repository.
- **depth** (*int*) – The Yum `repodata` depth. Usually 1.
- **kwargs** – see *Repository*

RECIPES = ('yum',)

configuration

As per *Repository.configuration* but specific to this repository recipe and type.

Return type str

nexuscli.api.repository.upload module

Methods to implement upload for specific repository formats (recipes)

nexuscli.api.repository.upload.**upload_file_raw**(*repository*, *src_file*, *dst_dir*, *dst_file*)

Upload a single file to a raw repository.

Parameters

- **repository** (nexuscli.api.repository.model.Repository) – repository instance used to access Nexus 3 service.
- **src_file** – path to the local file to be uploaded.
- **dst_dir** – directory under dst_repo to place file in. When None, the file is placed under the root of the raw repository
- **dst_file** – destination file name.

Raises

- ***exception.NexusClientInvalidRepositoryPath*** – invalid repository path.
- ***exception.NexusClientAPIError*** – unknown response from Nexus API.

nexuscli.api.repository.upload.**upload_file_yum**(*repository*, *src_file*, *dst_dir*, *dst_file*)
Upload a single file to a yum repository.

Parameters

- **repository** (nexuscli.api.repository.model.Repository) – repository instance used to access Nexus 3 service.
- **src_file** – path to the local file to be uploaded.
- **dst_dir** – directory under dst_repo to place file in.
- **dst_file** – destination file name.

Raises ***exception.NexusClientAPIError*** – unknown response from Nexus API.

nexuscli.api.repository.util module

nexuscli.api.repository.util.**get_files**(*src_dir*, *recurse=True*)
Walks the given directory and collects files to be uploaded. If recurse option is False, only the files on the root of the directory will be returned.

Parameters

- **src_dir** – location of files
- **recurse** – If false, only the files on the root of src_dir are returned

Returns file set to be used with upload_directory

Return type set

nexuscli.api.repository.util.**get_upload_subdirectory**(*dst_dir*, *file_path*, *flatten=False*)
Find the destination subdirectory based on given parameters. This is mostly so the *flatten* option is honoured.

Parameters

- **dst_dir** – destination directory
- **file_path** – file path, using REMOTE_PATH_SEPARATOR as the directory separator.
- **flatten** (*bool*) – when True, sub_directory will be flattened (ie: file_path structure will not be present in the destination directory)

Returns the appropriate sub directory in the destination directory.

Return type str

nexuscli.api.repository.validations module

nexuscli.api.repository.validations.**ensure_known**(*target*, *value*, *known*)
Validate whether the a target argument is known and supported. The `target` is only used to provide a friendlier message to the user. The given `value` is checked against `known` and `supported`.

Parameters

- **target** (*str*) – name of the target, as known to the end-user.

- **value** (*str*) – value of the target key.
- **known** (*list*, *tuple*) – known possible values for the target.

Raises **ValueError** – if given value is not in `known`.

Module contents

nexuscli.api.script package

Submodules

nexuscli.api.script.model module

class nexuscli.api.script.model.**Script**
Bases: object

A Class representing a Nexus 3 script.

class nexuscli.api.script.model.**ScriptCollection**(*client=None*)
Bases: object

A class to manage Nexus 3 scripts.

Parameters **client** (nexuscli.nexus_client.NexusClient) – the client instance that will be used to perform operations against the Nexus 3 service. You must provide this at instantiation or set it before calling any methods that require connectivity to Nexus.

client
as per `client` argument of *ScriptCollection*.

Type *nexuscli.nexus_client.NexusClient*

create(*script_name*, *script_content*, *script_type='groovy'*)
Create the given script in the Nexus 3 service.

Parameters

- **script_name** (*str*) – name of script to be created.
- **script_content** (*str*) – script code.
- **script_type** (*str*) – type of script to be created.

Raises ***exception.NexusClientAPIError*** – if the script creation isn't successful; i.e.: any HTTP code other than 204.

create_if_missing(*name*, *content=None*, *script_type='groovy'*)
Creates a script in the Nexus 3 service IFF a script with the same name doesn't exist. Equivalent to checking if the script exists with *get()* and, if not, creating it with *create()*.

Parameters

- **name** (*str*) – name of script to be created.
- **content** (*Union[str,NoneType]*) – script code. If not given, the method will use *nexuscli.nexus_util.groovy_script()* to read the script code from a local file.
- **script_type** (*str*) – type of script to be created.

Raises ***exception.NexusClientAPIError*** – if the script creation isn't successful; i.e.: any HTTP code other than 204.

delete(*script_name*)
Deletes a script from the Nexus 3 repository.

Parameters **script_name** – name of script to be deleted.

Raises ***exception.NexusClientAPIError*** – if the Nexus service fails to delete the script; i.e.: any HTTP code other than 204.

exists(*name*)
Check if a script exists.

Parameters **name** – of script to verify existence.

Returns True if it exists, false otherwise

Return type bool

Raises ***exception.NexusClientAPIError*** – if the response from the Nexus service isn't recognised; i.e.: any HTTP code other than 200, 404.

get(*name*)
Get a Nexus 3 script by name.

Parameters **name** – of script to be retrieved.

Returns the script or None, if not found

Return type dict, None

Raises ***exception.NexusClientAPIError*** – if the response from the Nexus service isn't recognised; i.e.: any HTTP code other than 200, 404.

list()
List of all script names on the Nexus 3 service.

Returns a list of names

Return type list

Raises ***exception.NexusClientAPIError*** – if the script names cannot be retrieved; i.e.: any HTTP code other than 200.

run(*script_name*, *data=''*)
Runs an existing script on the Nexus 3 service.

Parameters

- **script_name** – name of script to be run.
- **data** – parameters to be passed to the script, via HTTP POST. If the script being run requires a certain format or encoding, you need to prepare it yourself. Typically this is *json.dumps(data)*.

Returns the content returned by the script, if any.

Return type str, dict

Raises ***exception.NexusClientAPIError*** – if the Nexus service fails to run the script; i.e.: any HTTP code other than 200.

Module contents

Module contents

nexuscli.cli package

Submodules

nexuscli.cli.errors module

class nexuscli.cli.errors.**CliReturnCode**
Bases: enum.Enum

Error codes returned by *nexuscli.cli*

API_ERROR = 2

CONNECTION_ERROR = 3

INVALID_SUBCOMMAND = 10

NO_FILES = 1

POLICY_NOT_FOUND = 20

REPOSITORY_NOT_FOUND = 30

SUCCESS = 0

UNKNOWN_ERROR = 99

nexuscli.cli.root_commands module

Handles base/root commands (as opposed to subcommands)

nexuscli.cli.root_commands.**cmd_del**(**args*, ***kwargs*)
Alias for *cmd_delete()*

nexuscli.cli.root_commands.**cmd_delete**(*nexus_client*, *options*)
Performs `nexus3 delete`

nexuscli.cli.root_commands.**cmd_dl**(**args*, ***kwargs*)
Alias for *cmd_download()*

nexuscli.cli.root_commands.**cmd_download**(*nexus_client*, *args*)
Performs `nexus3 download`

nexuscli.cli.root_commands.**cmd_list**(*nexus_client*, *args*)
Performs `nexus3 list`

nexuscli.cli.root_commands.**cmd_login**(*_*, *__*)
Performs `nexus3 login`

nexuscli.cli.root_commands.**cmd_ls**(**args*, ***kwargs*)
Alias for *cmd_list()*

nexuscli.cli.root_commands.**cmd_up**(**args*, ***kwargs*)
Alias for *cmd_upload()*

nexuscli.cli.root_commands.**cmd_upload**(*nexus_client*, *args*)
Performs `nexus3 upload`

nexuscli.cli.subcommand_cleanup_policy module

Usage: nexus3 cleanup_policy –help nexus3 cleanup_policy create <policy_name> [–format=<format>]

[–downloaded=<days>] [–updated=<days>]

nexus3 cleanup_policy list

Options: -h –help This screen –format=<format> Accepted: all or a repository format [default: all] –downloaded=<days> Cleanup criteria; last downloaded in this many days. –updated=<days> Cleanup criteria; last updated in this many days.

Commands: cleanup_policy create Create or update the cleanup policy <policy_name> cleanup_policy list List all existing cleanup policies.

nexuscli.cli.subcommand_cleanup_policy.**cmd_create**(*nexus_client*, *args*)
Performs `nexus3 cleanup_policy create`

nexuscli.cli.subcommand_cleanup_policy.**cmd_list**(*nexus_client*, *_*)
Performs `nexus3 cleanup_policy list`

nexuscli.cli.subcommand_cleanup_policy.**main**(*argv=None*)
Entrypoint for `nexus3 cleanup_policy` subcommand.

nexuscli.cli.subcommand_repository module

Usage: nexus3 repository –help nexus3 repository list nexus3 repository show <repo_name> nexus3 repository (delete|del) <repo_name> [–force] nexus3 repository create hosted (bower|npm|nuget|pypi|raw|rubygems)

<repo_name> [–blob=<store_name>] [–strict-content] [–cleanup=<c_policy>] [–write=<w_policy>]

nexus3 repository create proxy (bower|npm|nuget|pypi|raw|rubygems|yum) <repo_name> <remote_url> [–blob=<store_name>] [–strict-content] [–cleanup=<c_policy>]

nexus3 repository create hosted maven <repo_name> [–blob=<store_name>] [–strict-content] [–cleanup=<c_policy>] [–write=<w_policy>] [–version=<v_policy>] [–layout=<l_policy>]

nexus3 repository create proxy maven <repo_name> <remote_url> [–blob=<store_name>] [–strict-content] [–cleanup=<c_policy>] [–version=<v_policy>] [–layout=<l_policy>]

nexus3 repository create hosted yum <repo_name> [–blob=<store_name>] [–strict-content] [–cleanup=<c_policy>] [–write=<w_policy>] [–depth=<repo_depth>]

Options: -h –help This screen –blob=<store_name> Use this blob with new repository [default: default] –depth=<repo_depth> Depth (0-5) where repodata folder(s) exist [default: 0] –layout=<l_policy> Accepted: strict, permissive [default: strict] –strict-content Enable strict content type validation –version=<v_policy> Accepted: release, snapshot, mixed [default: release] –write=<w_policy> Accepted: allow, allow_once, deny [default: allow_once] –cleanup=<c_policy> Accepted: an existing Cleanup Policy name -f –force Do not ask for confirmation before deleting

Commands: repository create Create a repository using the format and options provided repository delete Delete a repository. repository list List all repositories available on the server repository show Show the configuration for a repository as JSON.

nexuscli.cli.subcommand_repository.**cmd_create**(*nexus_client*, *args*)
Performs `nexus3 repository create` commands

nexuscli.cli.subcommand_repository.**cmd_del**(**args*, ***kwargs*)
Alias for *cmd_delete()*

nexuscli.cli.subcommand_repository.**cmd_delete**(*nexus_client*, *args*)
Performs `nexus3 repository delete`

nexuscli.cli.subcommand_repository.**cmd_list**(*nexus_client*, *_*)
Performs `nexus3 repository list`

nexuscli.cli.subcommand_repository.**cmd_show**(*nexus_client*, *args*)
Performs “nexus3 repository show

nexuscli.cli.subcommand_repository.**main**(*argv=None*)
Entrypoint for `nexus3 repository` subcommand.

nexuscli.cli.subcommand_script module

Usage: nexus3 script –help nexus3 script create <script_name> <script_path> [–script_type=<type>] nexus3 script list nexus3 script run <script_name> [<script_args>] nexus3 script (delete|del) <script_name>

Options: -h –help This screen –script_type=<type> Script type [default: groovy]

Commands: script create Create or update a script using the <script_path> file script list List all scripts available on the server script del Remove existing <script_name> script run Run the existing <script_name> with optional <script_args>

nexuscli.cli.subcommand_script.**cmd_create**(*nexus_client*, *args*)
Performs `nexus3 script create`

nexuscli.cli.subcommand_script.**cmd_del**(**args*, ***kwargs*)
Alias for *cmd_delete()*

nexuscli.cli.subcommand_script.**cmd_delete**(*nexus_client*, *args*)
Performs `nexus3 script delete`

nexuscli.cli.subcommand_script.**cmd_list**(*nexus_client*, *_*)
Performs `nexus3 script list`

nexuscli.cli.subcommand_script.**cmd_run**(*nexus_client*, *args*)
Performs `nexus3 script run`

nexuscli.cli.subcommand_script.**main**(*argv=None*)
Entrypoint for `nexus3 script` subcommand.

nexuscli.cli.util module

nexuscli.cli.util.**find_cmd_method**(*arguments*, *methods*)
Helper to find the corresponding python method for a CLI command.

Suitable python methods must be named `cmd_COMMAND`, where *COMMAND* is the CLI command and `cmd_` is a hard-coded prefix.

Parameters

- **arguments** – the return of `docopt.docopt()`.
- **methods** – the return value from *globals()*, as-is

Returns the python method corresponding to the given CLI command. None if no suitable method is found.

Return type Union[callable, None]

nexuscli.cli.util.**get_client**()
Returns a Nexus Client instance. Prints a warning if a configuration file isn't file.

Return type *nexuscli.nexus_client.NexusClient*

nexuscli.cli.util.**input_with_default**(*prompt*, *default=None*)
Prompts for a text answer with an optional default choice.

Parameters

- **prompt** – question to be displayed to user
- **default** – default choice

Returns user-provided answer or None, if default not provided.

Return type Union[str,None]

Module contents

Nexus 3 CLI.

Usage: nexus3 –help # run this to see full list of commands/subcommands nexus3 –version nexus3 login nexus3 (list|ls) <repository_path> nexus3 (upload|up) <from_src> <to_repository> [–flatten] [–norecurse] nexus3 (download|dl) <from_repository> <to_dst> [–flatten] [–nocache] nexus3 (delete|del) <repository_path> nexus3 <subcommand> [<arguments>. . .]

Options:

-h –help This screen. For help with sub-commands, run *nexus3 <subcommand> -h*

--version	Show the Nexus3 CLI version and exit
--flatten	Flatten directory structure on *nexus3* transfers [default: False]
--nocache	Force download even if local copy is up-to-date [default: False]
--norecurse	Don't process subdirectories on *nexus3 up* transfers [default: False]

Commands: login Test login and save credentials to ~/.nexus-cli list List all files within a path in the repository upload Upload file(s) to designated repository download Download an artefact or a directory to local file system delete Delete artefact(s) from repository

Sub-commands: cleanup_policy Cleanup Policy management. repository Repository management. script Script management.

nexuscli.cli.**main**(*argv=None*)
Entrypoint for the setuptools CLI console script

Submodules

nexuscli.exception module

exception nexuscli.exception.**DownloadError**
Bases: Exception

Error retrieving artefact from Nexus service.

exception nexuscli.exception.**NexusClientAPIError**
Bases: Exception

Unexpected response from Nexus service.

exception nexuscli.exception.**NexusClientConnectionError**
Bases: Exception

Generic network connector error.

exception nexuscli.exception.**NexusClientCreateCleanupPolicyError**
Bases: Exception

Used when a cleanup policy creation operation in Nexus fails.

exception nexuscli.exception.**NexusClientCreateRepositoryError**
Bases: Exception

Used when a repository creation operation in Nexus fails.

exception nexuscli.exception.**NexusClientInvalidCleanupPolicy**
Bases: Exception

The given cleanup policy does not exist in Nexus.

exception nexuscli.exception.**NexusClientInvalidCredentials**
Bases: Exception

Login credentials not accepted by Nexus service. Usually the result of a HTTP 401 response.

exception nexuscli.exception.**NexusClientInvalidRepository**
Bases: Exception

The given repository does not exist in Nexus.

exception nexuscli.exception.**NexusClientInvalidRepositoryPath**
Bases: Exception

Used when an operation against the Nexus service uses an invalid or non-existent path.

nexuscli.nexus_client module

class nexuscli.nexus_client.**NexusClient**(*config=None*)
Bases: object

A class to interact with Nexus 3's API.

Unless all keyword arguments `url`, `user` and `password` are supplied, the class will attempt to read the configuration file and, if unsuccessful, use defaults.

Parameters **config** (NexusConfig) – instance containing the configuration for the Nexus service used by this instance.

cleanup_policies
Instance of *CleanupPolicyCollection* . This will automatically use the existing instance of *NexusClient* to communicate with the Nexus service.

delete(*repository_path*)
Delete artefacts, recursively if `repository_path` is a directory.

Parameters **repository_path** (*str*) – location on the repository service.

Returns number of deleted files. Negative number for errors.

Return type int

download(*source*, *destination*, *flatten=False*, *nocache=False*)
Process a download. The source must be a valid Nexus 3 repository path, including the repository name as the first component of the path.

The destination must be a local file name or directory.

If a file name is given as destination, the asset may be renamed. The final destination will depend on `flatten`.

Parameters

- **source** (*str*) – location of artefact or directory on the repository service.
- **destination** (*str*) – path to the local file or directory.
- **flatten** (*bool*) – if True, the remote path isn't reproduced locally.
- **nocache** (*bool*) – if True, force download of a directory or artefact, ignoring an existing local copy. If false, it will not re-download an existing copy if its checksum matches the one in Nexus (as determined by *nexuscli.nexus_util.has_same_hash()*).

Returns number of downloaded files.

Return type int

download_file(*download_url*, *destination*)
Download an asset from Nexus artefact repository to local file system.

Parameters

- **download_url** (*str*) – fully-qualified URL to asset being downloaded.
- **destination** (*str*) – file or directory location to save downloaded asset. Must be an existing directory; any exiting file in this location will be overwritten.

Returns

http_delete(*endpoint*, ***kwargs*)
Performs a HTTP DELETE request on the given endpoint.

Parameters

- **endpoint** (*str*) – name of the Nexus REST API endpoint.
- **kwargs** – as per `requests.request()`.

Return type requests.Response

http_get(*endpoint*)
Performs a HTTP GET request on the given endpoint.

Parameters **endpoint** (*str*) – name of the Nexus REST API endpoint.

Return type requests.Response

http_head(*endpoint*)
Performs a HTTP HEAD request on the given endpoint.

Parameters **endpoint** (*str*) – name of the Nexus REST API endpoint.

Return type requests.Response

http_post(*endpoint*, ***kwargs*)
Performs a HTTP POST request on the given endpoint.

Parameters

- **endpoint** (*str*) – name of the Nexus REST API endpoint.

- **kwargs** – as per `requests.request()`.

Return type requests.Response

http_put(*endpoint*, ***kwargs*)
Performs a HTTP PUT request on the given endpoint.

Parameters

- **endpoint** (*str*) – name of the Nexus REST API endpoint.
- **kwargs** – as per `requests.request()`.

Return type requests.Response

http_request(*method*, *endpoint*, *service_url=None*, ***kwargs*)
Performs a HTTP request to the Nexus REST API on the specified endpoint.

Parameters

- **method** – one of `get`, `put`, `post`, `delete`.
- **endpoint** (*str*) – URI path to be appended to the service URL.
- **service_url** (*str*) – override the default URL to use for the request, which is created by joining *rest_url* and `endpoint`.
- **kwargs** – as per `requests.request()`.

Return type requests.Response

list(*repository_path*)
List all the artefacts, recursively, in a given `repository_path`.

Parameters **repository_path** (*str*) – location on the repository service.

Returns artefacts under `repository_path`.

Return type typing.Iterator[str]

list_raw(*repository_path*)
As per *list()* but yields raw Nexus artefacts as dicts.

Parameters **repository_path** (*str*) – location on the repository service.

Return type typing.Iterator[dict]

repositories
Instance of *RepositoryCollection*. This will automatically use the existing instance of *NexusClient* to communicate with the Nexus service.

Return type *RepositoryCollection*

rest_url
Full URL to the Nexus REST API, based on the `url` and `version` from `config`.

Return type str

scripts
Instance of *ScriptCollection*. This will automatically use the existing instance of *NexusClient* to communicate with the Nexus service.

server_version
Parse the Server header from a Nexus request response and return as version information. The method expects the header Server to be present and formatted as, e.g., 'Nexus/3.19.1-01 (OSS)'

Returns the parsed version. If it can't be determined, return None.

Return type Union[None,semver.VersionInfo]

split_component_path(*component_path*)

Splits a given component path into repository, directory, filename.

A Nexus component path for a raw directory must have this format:

```
repository_name/directory[(/subdir1)...][/|filename]
```

A path ending in / represents a directory; otherwise it represents a filename.

```
>>> dst0 = 'myrepo0/dir/'
>>> dst1 = 'myrepo1/dir/subdir/'
>>> dst2 = 'myrepo2/dir/subdir/file'
>>> dst3 = 'myrepo3/dir/subdir/etc/file.ext'
>>> split_component_path(dst0)
>>> ('myrepo0', 'dir', None)
>>> split_component_path(dst1)
>>> ('myrepo1', 'dir/subdir', None)
>>> split_component_path(dst2)
>>> ('myrepo2', 'dir/subdir', 'file')
>>> split_component_path(dst3)
>>> ('myrepo3', 'dir/subdir/etc', 'file.ext')
```

Parameters **component_path** (*str*) – the Nexus component path, as described above.

Returns tuple of `(repository_name, directory, filename)`. If the given `component_path` doesn't represent a file, then the `filename` is set to `None`.

Return type tuple[str, str, str]

upload(*source*, *destination*, *recurse=True*, *flatten=False*)

Process an upload. The source must be either a local file name or directory. The flatten and recurse options are honoured for directory uploads.

The destination must be a valid Nexus 3 repository path, including the repository name as the first component of the path.

Parameters

- **source** (*str*) – location of file or directory to be uploaded.
- **destination** (*str*) – destination path in Nexus, including repository name and, if required, directory name (e.g. raw repos require a directory).
- **recurse** (*bool*) – do not process sub directories for uploads to remote
- **flatten** (*bool*) – Flatten directory structure by not reproducing local directory structure remotely

Returns number of files uploaded.

nexuscli.nexus_config module

class nexuscli.nexus_config.**NexusConfig**(*username='admin'*, *password='admin123'*, *url='http://localhost:8081'*, *x509_verify=True*, *api_version='v1'*, *config_path=None*)

Bases: object

A class to hold Nexus 3's configuration.

Unless keyword arguments `url`, `user` and `password` are supplied, the class will attempt to read the configuration file and, if unsuccessful, use defaults.

Parameters

- **username** (*str*) – username for Nexus service at given url.
- **password** (*str*) – password for username above.
- **url** (*str*) – URL to Nexus 3 OSS service.
- **x509_verify** (*bool*) – toggle certificate validation.
- **api_version** (*str*) – Nexus REST API version to be used.
- **config_path** (*str*) – local file containing configuration above in JSON format with these keys: `nexus_url`, `nexus_user`, `nexus_pass` and `nexus_verify`.

api_version
Current API version in use.

Return type str

auth
Current username and password as a tuple.

Return type tuple[str, str]

config_file
Path to configuration file, as given by `config_path` during instantiation.

Return type str

dump()
Writes the current configuration to disk under property:*config_file*.

If a file already exists, it will be overwritten. The permission will be set to read/write to the owner only.

load()
Load the configuration settings from the file specified by *config_file*.

The configuration file is in JSON format and expects these keys: `nexus_user`, `nexus_pass`, `nexus_url`, `nexus_verify`.

to_dict
Current instance configuration.

Return type dict

url
The Nexus service URL

Return type str

x509_verify
Whether to validate the x509 certificate when using https to access the Nexus service

Return type str

nexuscli.nexus_util module

nexuscli.nexus_util.**calculate_hash**(*hash_name*, *file_path_or_handle*)
Calculate a hash for the given file.

Parameters

- **hash_name** (*str*) – name of the hash algorithm in hashlib
- **file_path_or_handle** (*str*) – source file name (`str`) or file handle (`file-like`) for the hash algorithm.

Returns the calculated hash

Return type str

`nexuscli.nexus_util.`**ensure_exists**(*path*, *is_dir=False*)

Ensures a path exists.

Parameters

- **path** (*pathlib.Path*) – the path to ensure
- **is_dir** (*bool*) – whether the path is a directory.

`nexuscli.nexus_util.`**filtered_list_gen**(*raw_response*, *term=None*, *partial_match=True*)

Iterates over items yielded by raw_response_gen, validating that:

1. the *path* dict key is a str
2. the *path* value starts with starts_with (if provided)

```
>>> r = [{
>>>     'checksum': {
>>>         'md5': 'd94b865aa7620c46ef8faef7059a311c',
>>>         'sha1': '2186934d880cf24dd9ecc578335e290026695522',
>>>         'sha256': 'b7bb3424a6a6(...)4113bc38fd7807528481a8ffe3cf',
>>>         'sha512': 'e7806f3caa3e(...)3caeb9bbc54bbde286c07f837fdc'
>>>     },
>>>     'downloadUrl': 'http://nexus/repository/repo_name/a/file.ext',
>>>     'format': 'yum',
>>>     'id': 'Y2xvdWRlcmEtbWFuYWdlcj(...)mRiNWU0YjllZWQzMg',
>>>     'path': 'a/fake.rpm',
>>>     'repository': 'cloudera-manager'}]
>>>
>>> for i in filtered_list_gen(r, starts_with='a/fake.rpm')
>>>     print(i['path'])
a/fake.rpm
>>> for i in filtered_list_gen(r, starts_with='b')
>>>     print(i['path'])
# (nothing printed)
```

Parameters

- **raw_response** (*iterable*) – an iterable that yields one element of a nexus search response at a time, such as the one returned by _paginate_get().
- **term** (*str*) – if defined, only items with an attribute *path* that starts with the given parameter are returned.
- **partial_match** (*bool*) – if True, include items whose artefact path starts with the given term.

Yields *dict* – items that matched the filter.

`nexuscli.nexus_util.`**groovy_script**(*script_name*)

Returns the content for a groovy script located in the package installation path under script/groovy.

E.g.: groovy_script('foo') returns the content for the file at `.../site-packages/nexuscli/script/groovy/foo.groovy`.

Parameters **script_name** – file name of the groovy script; `.groovy` is appended to this parameter to form the file name.

Returns content for the groovy script

Return type str

`nexuscli.nexus_util.`**has_same_hash**(*artefact*, *filepath*)

Checks if a Nexus artefact has the same hash as a local filepath.

Parameters

- **artefact** (*dict*) – as returned by *list_raw()*
- **filepath** – local file path

Returns True if artefact and filepath have the same hash.

Return type bool

`nexuscli.nexus_util.`**validate_strings**(**args*)

Checks that all given arguments have a string type (e.g. str, basestring, unicode etc)

Parameters ***args** – values to be validated.

Returns True if all arguments are of a string type. False otherwise.

Return type bool

Module contents

2.2 Examples

Here are some basic operations to get you started. The CLI implementation in `src/nexuscli/cli.py` is another good source of examples.

In all examples below you will need to instantiate a client:

```
>>> import nexuscli
>>> nexus_config = nexuscli.nexus_config.NexusConfig(password='supersecret')
>>> nexus_client = nexuscli.nexus_client.NexusClient(config=nexus_config)
>>> # update the local list of repositories on the client
>>> nexus_client.repositories.refresh()
>>> # retrieve the list of repositories
>>> repositories = nexus_client.repositories.raw_list()
>>> repositories[0]
{'name': 'maven-snapshots',
 'format': 'maven2',
 'type': 'hosted',
 'url': 'http://localhost:8081/repository/maven-snapshots'}
```

2.2.1 Create a repository

```
>>> r = nexuscli.repository.Repository(
>>>     'hosted',
>>>     name='my-repository',
>>>     format='raw',
>>>     blob_store_name='default',
>>>     strict_content_type_validation=False,
>>>     write_policy='allow',
>>> )
>>> nexus_client.repositories.create(r)
>>> nexus_client.repositories.get_raw_by_name('my-repository')
{'name': 'my-repository',
 'format': 'raw',
 'type': 'hosted',
 'url': 'http://localhost:8081/repository/my-repository'}
```

2.2.2 Delete a repository

```
>>> nexus_client.repositories.delete('my-repository')
```

2.2.3 Upload a file

```
>>> repository = nexus_client.repositories.get_by_name('my-repository')
>>> upload_count = repository.upload('/etc/passwd', '/etc/passwd')
>>> print(upload_count)
1
```

2.3 Sonatype Nexus 3 API documentation

- REST API
- Script API

CHAPTER 3

Groovy Scripts

This package makes use of Groovy scripts to perform actions that are not available through the Nexus 3 REST API.

All scripts added have names starting with `nexus3-cli-`.

```
$ nexus3 script list
Name (type)
nexus3-cli-cleanup-policy (groovy)
nexus3-cli-repository-create (groovy)
```

You can delete them all by running:

```
$ nexus3 script list | awk '{ print $1 }' | xargs --no-run-if-empty -n1 nexus3␣
↪script del
Name (type)
```

To increase verbosity of logging for the scripts, create a new logger (e.g.: http://localhost:8081/#admin/support/logging) with logger name `org.sonatype.nexus.script.plugin.internal.rest.ScriptResource` and logging level `DEBUG` or `TRACE`.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

n

Index

D

E

F

G

H

I

L

M

N

P

R

S

T

U

V

W

X

Y