rtpy Documentation

Guillaume Renault

Jul 06, 2020

Contents

CHAPTER 1

Installation

`rtpy` is compatible with Python 2.7 and 3.4+.

Use **`pip`** to install the latest stable version of `rtpy`:

```
$ pip install rtpy
```

CHAPTER 2

Getting started

2.1 Instantiate a rtpy.Rtpy object

A rtpy.Rtpy object is used to make all the API calls. To be instantiated the rtpy.Rtpy class only takes a **Python dictionary as first positional argument**. This dictionary contains the user's settings such as API key and Artifactory instance URL.

2.1.1 Mandatory keys

- **"af_url"** : URL of the AF instance (starting with http(s)://)
- **"api_key"** or **"username"** and **"password"** : API key or username and password for the user in the Artifactory instance

```
import rtpy

# instantiate a rtpy.Rtpy object
settings = {}
settings["af_url"] = "http://..."
settings["api_key"] = "123QWA..."
# settings["username"] = "my_username"
# settings["password"] = "my_password"

af = rtpy.Rtpy(settings)

# use a method
r = af.system_and_configuration.system_health_ping()
print(r)
# OK
```

2.1.2 Optional keys

- **"verbose_level"** : 0/1
  - The desired verbose level, 0 for nothing, 1 to print performed operations
  - 0 if not not provided
- **"raw_response"** : False/True
  - True will return a requests.Response object and the errors will not be automatically raised
  - False will return a python object
  - False if not provided
- **"session"**: requests.Session object
  - rtpy uses a requests.Session object to make calls to the Artifactory API endpoint. A custom can be provided session object when creating a rtpy.Rtpy object for advanced HTTP configurations, proxies, SSL...
  - request.Session() if not provided

```
import requests
import rtpy

settings["verbose_level"] = 0/1
settings["raw_response"] = False/True

# SSL : custom CA bundle example
session = requests.Session()
session.verify = "path/to/ca_bundle.crt"
settings['session'] = session

af = rtpy.Rtpy(settings)

r = af.system_and_configuration.system_health_ping()
print(r)
# OK
```

CHAPTER 3

Examples

List of examples for supported API methods.

3.1 ARTIFACTS AND STORAGE

3.1.1 Folder Info

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-FolderInfo

```
r = af.artifacts_and_storage.folder_info(repo_key, folder_path)
# repo_key is the name of the repository in Artifactory
# folder_path is the path of the folder inside the repo
# To get information on the root repo, use "", as argument for folder_path
```

3.1.2 File Info

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-FileInfo

```
r = af.artifacts_and_storage.file_info(repo_key, file_path)
# repo_key is the name of the repository in Artifactory
# file path is the path of the file inside the repo
```

3.1.3 Get Storage Summary Info

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetStorageSummaryInfo

```
r = af.artifacts_and_storage.get_storage_summary_info()
```

3.1.4 Item Last Modified

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-ItemLastModified

```
r = af.artifacts_and_storage.item_last_modified(repo_key, item_path)
# repo_key is the name of the repository in Artifactory
# item_path is the path of the item inside the repo
```

3.1.5 File Statistics

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-FileStatistics

```
r = af.artifacts_and_storage.file_statistics(repo_key, item_path)
# repo_key is the name of the repository in Artifactory
# item_path is the path of the item inside the repo
```

3.1.6 Item Properties

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-ItemProperties

```
r = af.artifacts_and_storage.item_properties(repo_key, item_path)
# repo_key is the name of the repository in Artifactory
# item_path is the path of the item inside the repo

# Standard example
r = af.artifacts_and_storage.item_properties("my_repo", "folder/artifact.png")

# Retrieve specific propertie(s)
r = af.artifacts_and_storage.item_properties(repo_key, item_path, properties="version
↪")
r = af.artifacts_and_storage.item_properties(repo_key, item_path, properties="version,
↪ owner")
```

3.1.7 Set Item Properties

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-SetItemProperties

```
r = af.artifacts_and_storage.set_item_properties(repo_key, item_path, properties)
# repo_key is the name of the repository in Artifactory
# item_path is the path of the item inside the repo

# Single property
r = af.artifacts_and_storage.set_item_properties(repo_key, item_path, "version=1.0")

# Set multiple properties
r = af.artifacts_and_storage.set_item_properties(repo_key, item_path, "version=1.0;
↪author=smith")

# Additionnal options from the documentation can be supplied as a string
r = af.artifacts_and_storage.set_item_properties(repo_key, item_path, "version=1.0;
↪author=smith", options=string_of_options)
# options_string is a string of the possible options
# Such as [&recursive=1]
```

3.1.8 Delete Item Properties

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeleteItemProperties

```
r = af.artifacts_and_storage.delete_item_properties(repo_key, item_path, properties)
# repo_key is the name of the repository in Artifactory
# item_path is the path of the item inside the repo

# Delete a single property
r = af.artifacts_and_storage.delete_item_properties(repo_key, item_path, "version")

# Delete multiple properties
r = af.artifacts_and_storage.delete_item_properties(repo_key, item_path, "version,
↪author")
```

3.1.9 Set Item SHA256 Checksum

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-SetItemSHA256Checksum

```
params = {"repo_key": my_repo_key, "path": mypath}
# repo_key is the name of the repository in Artifactory
# artifact_path is the path of the artifact inside the repo
r = af.artifacts_and_storage.set_item_sha256_checksum(params)
```

3.1.10 Retrieve Artifact

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-RetrieveArtifact

```
r = af.artifacts_and_storage.retrieve_artifact(repo_key, artifact_path)
# repo_key is the name of the repository in Artifactory
# artifact_path is the path of the artifact inside the repo

# Save the file locally
with open("myartifact.png", "wb") as artifact:
    artifact.write(r.content)
```

3.1.11 Retrieve Folder or Repository Archive

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-RetrieveFolderorRepositoryArchive

```
r = af.artifacts_and_storage.retrieve_folder_or_repository_archive(repo_key, path,␣
↪archive_type)
# repo_key is the name of the repository in Artifactory
# path is the path of the folder inside the repo
# archive_type can be "zip", "tar", "tar.gz", "tgz"

# Checksums can be included
r = af.artifacts_and_storage.retrieve_folder_or_repository_archive(repo_key, path,␣
↪archive_type, include_checksums=True)

# Save the archive locally
with open("myarchive.archive_type", "wb") as archive:
    archive.write(r.content)
```

3.1.12 Trace Artifact Retrieval

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-TraceArtifactRetrieval

```
r = af.artifacts_and_storage.trace_artifact_retrieval(repo_key, item_path)
# repo_key is the name of the repository in Artifactory
# item_path is the path of the item inside the repo

# with this method the response is a Python requests response object
# use r.text

print(r.text)
```

3.1.13 Create Directory

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CreateDirectory

```
r = af.artifacts_and_storage.create_directory(repo_key, directory_path)
# repo_key is the name of the repository in Artifactory
# directory_path is the path of the directory inside the repo

# Known issue : when trying to create a directory that already exists,
# response will not say already exist and nothing will happen.
```

3.1.14 Deploy Artifact

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeployArtifact

```
r = af.artifacts_and_storage.deploy_artifact(repo_key, local_artifact_path, target_
↪artifact_path)
# repo_key is the name of the repository in Artifactory
# target_artifact_path is the path of the artifact inside the repo
# local_artifact_path is the path of the artifact on the local machine

# Standard example
r = af.artifacts_and_storage.deploy_artifact("myrepo", "myartifact_on_my_machine.png",
↪ "directory/my_remote_artifact.png")

# It is possible to attach properties as part of deploying an artifact using
# Artifactory's Matrix Parameters :
# https://www.jfrog.com/confluence/display/RTF4X/
↪Using+Properties+in+Deployment+and+Resolution

# Single property
r = af.artifacts_and_storage.deploy_artifact("myrepo", "myartifact_on_my_machine",
↪"myartifact;prop1=value")

# Multiple properties
r = af.artifacts_and_storage.deploy_artifact("myrepo", "myartifact_on_my_machine",
↪"myartifact;prop1=value;prop2=value2")
```

3.1.15 Deploy Artifact by Checksum

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeployArtifactbyChecksum

```
r = af.artifacts_and_storage.deploy_artifact_by_checksum(repo_key, target_artifact_
↪path, sha_type, sha_value):
# repo_key is the name of the repository in Artifactory
# target_artifact_path is the path of the artifact inside the repo
# sha_type is "sha1" or "sha256"
# sha_value is the value of the sha (string)

# Standard example
sha_type = "sha1"
sha_value = "e1a13e64b0414015d43dd80eed7876d7cee5e50e"

r = af.artifacts_and_storage.deploy_artifact_by_checksum("my_repo", "my_remote_
↪artifact", sha_type, sha_value)


# It is possible to attach properties as part of deploying an artifact using
# Artifactory's Matrix Parameters :
# https://www.jfrog.com/confluence/display/RTF4X/
↪Using+Properties+in+Deployment+and+Resolution

# Single property
r = af.artifacts_and_storage.deploy_artifact_by_checksum("myrepo", "myartifact;
↪prop1=value", sha_type, sha_value)

# Multiple properties
r = af.artifacts_and_storage.deploy_artifact_by_checksum("myrepo", "myartifact;
↪prop1=value;prop2=value2", sha_type, sha_value)
```

3.1.16 Delete Item

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeleteItem

```
r = af.artifacts_and_storage.delete_item(repo_key, path_to_item)
# repo_key is the name of the repository in Artifactory
# path_to_item is the path to the item (repo or artifact) in the repo
# use "" as argument for path_to_item to delete all the content of a repository
```

3.1.17 Copy Item

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CopyItem

```
r = af.artifacts_and_storage.copy_item(src_repo_key, src_item_path, target_repo_key,␣
↪target_item_path)
# src_repo_key is the name of the repository in Artifactory
# src_item_path is the path to the item (repo or artifact) in the repo
# target_repo_key is the name of the target repository in Artifactory
# target_item_path is the path of the item in the target repository

# Additionnal options from the documentation can be supplied as a string
r = af.artifacts_and_storage.copy_item(src_repo_key, src_item_path, target_repo_key,␣
↪target_item_path, options=string_of_options)
# Such as "[&dry=1][&suppressLayouts=0/1(default)][&failFast=0/1]"
```

3.1.18 Move Item

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-MoveItem

```
r = af.artifacts_and_storage.move_item(src_repo_key, src_item_path, target_repo_key,␣
↪target_item_path)
# src_repo_key is the name of the repository in Artifactory
# src_item_path is the path to the item (repo or artifact) in the repo
# target_repo_key is the name of the target repository in Artifactory
# target_item_path is the path of the item in the target repository

# Additionnal options from the documentation can be supplied as a string
r = af.artifacts_and_storage.move_item(src_repo_key, src_item_path, target_repo_key,␣
↪target_item_path, options=string_of_options)
# Such as [&dry=1][&suppressLayouts=0/1(default)][&failFast=0/1]
```

3.1.19 Artifact Sync Download

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-ArtifactSyncDownload

```
r = af.artifacts_and_storage.artifact_sync_download(repo_key, artifact_path)
# repo_key is the name of the repository in Artifactory
# artifact_path is the path of the artifact inside the repo

# Additionnal options from the documentation can be supplied as a string
r = af.artifacts_and_storage.artifact_sync_download(repo_key, artifact_path,␣
↪options=string_of_options)
# Such as [?content=none/progress][&mark=numOfBytesToPrintANewProgressMark]
# If no content parameter is specified the file content is downloaded to the client.
```

3.1.20 File List

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-FileList

```
r = af.artifacts_and_storage.file_list(repo_key, folder_path)
# repo_key is the name of the repository in Artifactory
# folder_path is the path of the folder inside the repo
# To get information on the root repo, use "", as argument for folder_path

# Additionnal options from the documentation can be supplied as a string
r = af.artifacts_and_storage.file_list(repo_key, folder_path, options=string_of_
↪options)
# Such as [&depth=n][&listFolders=0/1][&mdTimestamps=0/1][&includeRootPath=0/1]
```

3.1.21 Get Background Tasks

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetBackgroundTasks

```
r = af.artifacts_and_storage.get_background_tasks()
```

3.1.22 Empty Trash Can

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-EmptyTrashCan

```
r = af.artifacts_and_storage.empty_trash_can()
```

3.1.23 Delete Item From Trash Can

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeleteItemFromTrashCan

```
r = af.artifacts_and_storage.delete_item_from_trash_can(path_in_trashcan)
# path_in_trashcan is the path of the item inside the trashcan, typically : repo_name/
↪folder/file
```

3.1.24 Restore Item From Trash Can

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-RestoreItemfromTrashCan

```
r = af.artifacts_and_storage.restore_item_from_trash_can(path_in_trashcan, target_
↪path)
# path_in_trashcan is the path of the item inside the trashcan, typically : repo_name/
↪folder/file
# target_path is the path where the item will be restored, repo_name/folder/file
```

3.1.25 Optimize System Storage

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-OptimizeSystemStorage

```
r = af.artifacts_and_storage.optimize_system_storage()
```

3.2 BUILDS

3.2.1 All Builds

```
r = af.builds.all_builds()
```

3.3 REPOSITORIES

3.3.1 Get Repositories

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetRepositories

```
r = af.repositories.get_repositories()

# Additionnal options from the documentation can be supplied as a string
r = af.repositories.get_repositories(options=string_of_options)
```

(continues on next page)

(continued from previous page)

```
# Such as [?type=repositoryType (local|remote|virtual|distribution)]
# [&
↪packageType=maven|gradle|ivy|sbt|helm|cocoapods|opkg|rpm|nuget|cran|gems|npm|bower|debian|composer
```

3.3.2 Repository Configuration

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-RepositoryConfiguration

```
r = af.repositories.repository_configuration(repo_key)
# repo_key is the name of the repository in Artifactory
```

3.3.3 Create Repository

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CreateRepository

```
params = {}
params["key"] = "my_repo_name"
params["rclass"] = "local"
params["packageType"] = "debian"
# for remote repos : params["url"] = "http://..."
# for virtual repos : params["repositories"] = ["repo1", "repo2"]
r = af.repositories.create_repository(params)
# params is a dictionary (some fields are mandatory) of the repository settings
# https://www.jfrog.com/confluence/display/RTF/Repository+Configuration+JSON
↪#RepositoryConfigurationJSON-application/vnd.org.jfrog.artifactory.repositories.
↪LocalRepositoryConfiguration+json
```

3.3.4 Update Repository Configuration

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-UpdateRepositoryConfiguration

```
params = {}
params["key"] = "my_repo_name"
params["description"] = "new_description"
r = af.repositories.update_repository_configuration(params)
# params is a dictionary (some fields are mandatory) of the repository settings that␣
↪will be updated
```

3.3.5 Delete Repository

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeleteRepository

```
r = af.repositories.delete_repository(repo_key)
# repo_key is the name of the repository in Artifactory
```

3.3.6 Calculate YUM Repository Metadata

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateYUMRepositoryMetadata

```
r = af.repositories.calculate_yum_repository_metadata(repo_key)
# repo_key is the name of the repository in Artifactory

# Additionnal options from the documentation can be supplied as a string
r = af.calculate_yum_repository_metadata(repo_key, options=string_of_options)
# Such as [?path={path to repodata dir][&async=0/1]

# a GPG passphrase can be supplied
gpg_passphrase = "abc"
r = af.calculate_yum_repository_metadata(repo_key, x_gpg_passphrase=gpg_passphrase)
```

3.3.7 Calculate NuGet Repository Metadata

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateNuGetRepositoryMetadata

```
r = af.repositories.calculate_nuget_repository_metadata(repo_key)
# repo_key is the name of the repository in Artifactory
```

3.3.8 Calculate Npm Repository Metadata

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateNpmRepositoryMetadata

```
r = af.repositories.calculate_npm_repository_metadata(repo_key)
# repo_key is the name of the repository in Artifactory
```

3.3.9 Calculate Maven Index

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateMavenIndex

```
r = af.repositories.calculate_maven_index(options)
# options is a string of the possible options
# Such as [?repos=x[,y]][&force=0/1]
```

3.3.10 Calculate Maven Metadata

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateMavenMetadata

```
r = af.repositories.calculate_maven_metadata(repo_key, folder_path)
# repo_key is the name of the repository in Artifactory
# folder_path is the path of the folder inside the repo

# Additionnal options from the documentation can be supplied as a string
r = af.repositories.calculate_maven_metadata(repo_key, folder_path, options=string_of_
↪options)
# Such as {nonRecursive=true | false}
```

3.3.11 Calculate Debian Repository Metadata

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateDebianRepositoryMetadata

```
r = af.repositories.calculate_debian_repository_metadata(repo_key)
# repo_key is the name of the repository in Artifactory

# Additionnal options from the documentation can be supplied as a string
r = af.calculate_debian_repository_metadata(repo_key, options=string_of_options)
# Such as [?async=0/1][?writeProps=0/1]

# a GPG passphrase can be supplied
gpg_passphrase = "abc"
r = af.calculate_debian_repository_metadata(repo_key, x_gpg_passphrase=gpg_passphrase)
```

3.3.12 Calculate Opkg Repository Metadata

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateOpkgRepositoryMetadata

```
r = af.repositories.calculate_opkg_repository_metadata(repo_key)
# repo_key is the name of the repository in Artifactory

# Additionnal options from the documentation can be supplied as a string
r = af.calculate_opkg_repository_metadata(repo_key, options=string_of_options)
# Such as [?async=0/1][?writeProps=0/1]

# a GPG passphrase can be supplied
gpg_passphrase = "abc"
r = af.calculate_opkg_repository_metadata(repo_key, x_gpg_passphrase=gpg_passphrase)
```

3.3.13 Calculate Bower Index

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateBowerIndex

```
r = af.repositories.calculate_bower_index(repo_key)
# repo_key is the name of the repository in Artifactory
```

3.3.14 Calculate Helm Chart Index

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateHelmChartIndex

```
r = af.repositories.calculate_helm_chart_index(repo_key)
# repo_key is the name of the repository in Artifactory
```

3.3.15 Calculate CRAN Repository Metadata

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateCRANRepositoryMetadata

```
r = af.repositories.calculate_cran_repository_metadata(repo_key)
# repo_key is the name of the repository in Artifactory

# Additionnal options from the documentation can be supplied as a string
r = af.repositories.calculate_cran_repository_metadata(repo_key, options=string_of_
↪options)
# Such as [?async=0/1]
```

3.3.16 Calculate Conda Repository Metadata

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CalculateCondaRepositoryMetadata

```
r = af.repositories.calculate_conda_repository_metadata(repo_key)
# repo_key is the name of the repository in Artifactory

# Additionnal options from the documentation can be supplied as a string
r = af.repositories.calculate_conda_repository_metadata(repo_key, options=string_of_
↪options)
# Such as [?async=0/1]
```

3.4 SEARCHES

3.4.1 Artifactory Query Language

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-ArtifactoryQueryLanguage(AQL))

```
query = "aql_querry_string"
r = af.searches.artifactory_query_language(query)
# Example : query = "items.find({"repo":{"$eq":"my-repo"}})"
```

3.4.2 List Docker Repositories

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-ListDockerRepositories

```
r = af.searches.list_docker_repositories(repo_key)
# repo_key is the name of the repository in Artifactory

# Additionnal options from the documentation can be supplied as a string
r = af.searches.list_docker_repositories(repo_key, options=string_of_options)
# Such as ?n=<n from the request>&last=<last tag value from previous response>
```

3.4.3 List Docker Tags

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-ListDockerTags

```
r = af.searches.list_docker_tags(repo_key, image_path)
# repo_key is the name of the repository/registry in Artifactory
# image_path is the path of the docker image in the repository/registry

# Additionnal options from the documentation can be supplied as a string
r = af.searches.list_docker_repositories(repo_key, image_path, options=string_of_
↪options)
# Such as ?n=<n from the request>&last=<last tag value from previous response>
```

3.5 SECURITY

3.5.1 Get Users

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetUsers

```
r = af.security.get_users()
```

3.5.2 Get User Details

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetUserDetails

```
r = af.security.get_user_details(username)
# username is the name of the user in Artifactory
```

3.5.3 Get User Encrypted Password

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetUserEncryptedPassword

```
r = af.security.get_user_encrypted_password()
```

3.5.4 Create or Replace User

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CreateorReplaceUser

```
params = {}
params["name"] = "my_username"
params["admin"] = "false"
params["email"] = "myuser@orange.com"
params["password"] = "password"
r = af.security.create_or_replace_user(params)
# username is the name of the user in Artifactory
# params ia a dictionary of desired fields to use to create the user and their␣
↪value(s)
# https://www.jfrog.com/confluence/display/RTF4X/Security+Configuration+JSON
```

3.5.5 Update User

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-UpdateUser

```
params = {}
params["admin"] = "true"
r = af.security.update_user(params)
# username is the name of the user in Artifactory
# params ia a dictionary of desired fields to update and their value(s)
# https://www.jfrog.com/confluence/display/RTF4X/Security+Configuration+JSON
```

3.5.6 Delete User

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeleteUser

```
r = af.security.delete_user(username)
# username is the name of the user in Artifactory.
```

3.5.7 Get Locked Out Users

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetLockedOutUsers

```
r = af.security.get_locked_out_users()
```

3.5.8 Unlock Locked Out User

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-UnlockLockedOutUser

```
r = af.security.unlock_locked_out_user(username)
# username is the name of the user in Artifactory.
```

3.5.9 Unlock Locked Out Users

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-UnlockLockedOutUsers

```
r = af.security.unlock_locked_out_users(user_list)
# user_list is a python list of the users to unlock
```

3.5.10 Unlock All Locked Out Users

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-UnlockAllLockedOutUsers

```
r = af.security.unlock_all_locked_out_users()
```

3.5.11 Create API Key

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CreateAPIKey

```
r = af.security.create_api_key()
```

3.5.12 Regenerate API Key

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-RegenerateAPIKey

```
r = af.security.regenerate_api_key()
```

3.5.13 Get API Key

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetAPIKey

```
r = af.security.get_api_key()
```

3.5.14 Revoke API Key

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-RevokeAPIKey

```
r = af.security.revoke_api_key()
```

3.5.15 Revoke User API Key

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-RevokeUserAPIKey

```
r = af.security.revoke_user_api_key(username)
# username is the name of the user in Artifactory
```

3.5.16 Get Groups

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetGroups

```
r = af.security.get_groups()
```

3.5.17 Get Group Details

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetGroupDetails

```
r = af.security.get_group_details(group_name)
# group_name is the name of the group in Artifactory.
```

3.5.18 Create or Replace Group

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CreateorReplaceGroup

```
params = {}
params["group_name"] = "my_group"
r = af.security.create_or_replace_group(params)
# params is a python dictionnary which should be like :
# https://www.jfrog.com/confluence/display/RTF4X/Security+Configuration+JSON
```

3.5.19 Update Group

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-UpdateGroup

```
params = {}
params["group_name"] = "my_group"
params["description"] = "my_description"
r = af.security.update_group(params)
# params is a python dictionnary which should be like :
# https://www.jfrog.com/confluence/display/RTF4X/Security+Configuration+JSON
```

3.5.20 Delete Group

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeleteGroup

```
r = af.security.delete_group(group_name)
# group_name is the name of the group in Artifactory.
```

3.5.21 Get Permission Targets

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetPermissionTargets

```
r = af.security.get_permission_targets()
```

3.5.22 Get Permission Target Details

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetPermissionTargetDetails

```
r = af.security.get_permission_target_details(permission_target_name)
# permission_target_name is the name of the permission in Artifactory.
```

3.5.23 Create or Replace Permission Target

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-CreateorReplacePermissionTarget

```
params = {}
params["name"] = "my_permission"
params["repositories"] = ["myrepo1", "myrepo2"]
r = af.security.create_or_replace_permission_target(params)
# https://www.jfrog.com/confluence/display/RTF4X/Security+Configuration+JSON
```

3.5.24 Delete Permission Target

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-DeletePermissionTarget

```
r = af.security.delete_permission_target(permission_target_name)
# permission_target_name is the name of the permission in Artifactory.
```

3.5.25 Effective Item Permissions

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-EffectiveItemPermissions

```
r = af.security.effective_item_permissions(repo_key, item_path)
# repo_key is the name of the repository in Artifactory
# item_path is the path of the item inside the repo
# To get information on the root repo, use "", as argument for item_path
```

3.6 SYSTEM AND CONFIGURATION

3.6.1 System Info

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-SystemInfo

```
r = af.system_and_configuration.system_info()
```

3.6.2 System Health Ping

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-SystemHealthPing

```
r = af.system_and_configuration.system_health_ping()
```

3.6.3 General Configuration

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GeneralConfiguration

```
r = af.system_and_configuration.general_configuration()
```

3.6.4 Save General Configuration

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-SaveGeneralConfiguration

```
r = af.system_and_configuration.save_general_configuration(xml_file_path)
# xml_file_path is the path on the local machine of the configuration file to be␣
↪pushed
```

3.6.5 License Information

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-LicenseInformation

```
r = af.system_and_configuration.license_information()
```

3.6.6 Install License

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-InstallLicense

```
params = {"licenseKey": "license_string"}
r = af.system_and_configuration.install_license(params)
```

3.6.7 Version and Addons Information

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-VersionandAdd-onsinformation

```
r = af.system_and_configuration.version_and_addons_information()
```

3.6.8 Get Reverse Proxy Configuration

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetReverseProxyConfiguration

```
r = af.system_and_configuration.get_reverse_proxy_configuration()
```

3.6.9 Get Reverse Proxy Configuration

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetReverseProxyConfiguration

```
r = af.system_and_configuration.get_reverse_proxy_configuration()
```

3.6.10 Get Reverse Proxy Snippet

https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-GetReverseProxySnippet

```
r = af.system_and_configuration.get_reverse_proxy_snippet()
```

CHAPTER 4

Responses

The response when using methods can be a :

- When “raw_response” is **False** (default):
  - python dictionary or list (a converted json output) (most cases)
  - unicode string ((requests.Response).text) (if the json content can’t be decoded/is missing/isn’t expected)
- When “raw_response” is **True** :
  - requests.Response Object

CHAPTER 5

Error handling

5.1 UserSettingsError

Standalone error, when the instantiation of the Rtpy object fails due to incorrect settings

```
try:
    af = rtpy.Rtpy(settings)
except rtpy.UserSettingsError:
    # Do stuff
```

5.2 AfApiError

When the status code is 4xx-5xx and the API sends a well formed JSON

The error has specific attributes

```
try:
    r = af.category.method_xyz()

except af.AfApiError as error:

    # All the attributes of the error
    print(dir(error))

    # Rtpy attributes for the error
    print(error.api_method)
    print(error.url)
    print(error.verb)
    print(error.status_code)
    print(error.message)
    print(error.print_message)
```

(continues on next page)

(continued from previous page)

```
    if error.status_code == 404:
        # Do stuff

    if error.status_code == 403:
        # Do stuff
```

5.3 MalformedAfApiError

When the status code is 4xx-5xx and the API sends a malformed JSON

```
try:
    # The JSON is currently malformed when the API sends an error when using this␣
↪method
    af.system_and_configuration.install_license(params)
except af.MalformedAfApiError:
   # Do stuff
```

5.4 RtpyError

When a method is called and parameters are missing or incorrect

```
try:
    # Providing "" for artifact_path will raise the RtpyError
    af.artifacts_and_storage.retrieve_artifact("repo_key", "")
except af.RtpyError:
    # Do stuff
```

CHAPTER 6

General tips

6.1 Verify connectivity with Rtpy self call

Call an instantiated Rtby object to verify connectivity (binding to the system health ping method)

```
import rtpy

# instantiate a Rtpy object
settings = {}
settings["af_url"] = "http://..."
settings["api_key"] = "123QWA..."
# settings["username"] = "my_username"
# settings["password"] = "my_password"

af = rtpy.Rtpy(settings)
r = af()
# or r = af.system_and_configuration.system_health_ping()
# print(r)
# OK
```

6.2 Environment variables as settings

Use environement variables for the api_key and af_url in the settings dictionary

```
settings = {}
settings["af_url"] = os.environ["ARTIFACTORY_URL"] # URL of the AF instance
settings["api_key"] = os.environ["ARTIFACTORY_API_KEY"] # User/Admin API key in the␣
↪given AF instance

af = rtpy.Rtpy(settings)
```

6.3 Overriding settings

All the settings can be overridden for a **single function call** (original settings are restored when the call is over) This is useful for debugging (verbose level) or not raising errors (raw_response). It can also be used to provide different credentials

```
r = af.category.method_xyz(settings={"raw_response" : True, "verbose_level" : 1})

r = af.category.method_xyz(settings={"verbose_level" : 1})

r = af.category.method_xyz(settings={"api_key" : "123ABC..."})

session = requests.Session()
session.verify = "path/to/ca_bundle.crt"
r = af.category.method_xyz(settings={"session" : session})
```

6.4 pretty-print

Use the pprint package to print the json responses in a more readable way

```
from pprint import pprint
...
r = af.category.method_xyz()
pprint(r)
```

6.5 Json file to Python dictionary

Convert a json file to a python dictionnary using the json_to_dict method

```
my_dict = rtpy.json_to_dict(json_file_path)
```

CHAPTER 7

Supported API methods

Original complete JFrog Artifactory REST API method list : https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API

7.1 BUILDS

- [x] All Builds
- [ ] Build Runs
- [ ] Build Upload
- [ ] Build Info
- [ ] Builds Diff
- [ ] Build Promotion
- [ ] Promote Docker Image
- [ ] Delete Builds
- [ ] Build Rename
- [ ] Push Build to Bintray
- [ ] Distribute Build
- [ ] Control Build Retention

7.2 ARTIFACTS AND STORAGE

- [x] Folder Info
- [x] File Info

- [x] Get Storage Summary Info
- [x] Item Last Modified
- [x] File Statistics
- [x] Item Properties
- [x] Set Item Properties
- [x] Delete Item Properties
- [x] Set Item SHA256 Checksum
- [x] Retrieve Artifact
- [ ] Retrieve Latest Artifact
- [ ] Retrieve Build Artifacts Archive
- [x] Retrieve Folder or Repository Archive
- [x] Trace Artifact Retrieval
- [ ] Archive Entry Download
- [x] Create Directory
- [x] Deploy Artifact
- [x] Deploy Artifact by Checksum
- [ ] Deploy Artifacts from Archive
- [ ] Push a Set of Artifacts to Bintray
- [ ] Push Docker Tag to Bintray
- [ ] Distribute Artifact
- [ ] File Compliance Info
- [x] Delete Item
- [x] Copy Item
- [x] Move Item
- [ ] Get Repository Replication Configuration
- [ ] Set Repository Replication Configuration
- [ ] Update Repository Replication Configuration
- [ ] Delete Repository Replication Configuration
- [ ] Scheduled Replication Status
- [ ] Pull/Push Replication
- [ ] Pull/Push Replication (Deprecated)
- [ ] Create or Replace Local Multi-push Replication
- [ ] Update Local Multi-push Replication
- [ ] Delete Local Multi-push Replication
- [ ] Enable or Disable Multiple Replications
- [ ] Get Global System Replication Configuration

- [ ] Block System Replication
- [ ] Unblock System Replication
- [x] Artifact Sync Download
- [ ] Folder Sync (Deprecated)
- [x] File List
- [x] Get Background Tasks
- [x] Empty Trash Can
- [x] Delete Item From Trash Can
- [x] Restore Item from Trash Can
- [x] Optimize System Storage
- [ ] Get Puppet Modules
- [ ] Get Puppet Module
- [ ] Get Puppet Releases
- [ ] Get Puppet Release

7.3 SEARCHES

- [x] Artifactory Query Language (AQL)
- [ ] Artifact Search (Quick Search)
- [ ] Archive Entries Search (Class Search)
- [ ] GAVC Search
- [ ] Property Search
- [ ] Checksum Search
- [ ] Bad Checksum Search
- [ ] Artifacts Not Downloaded Since
- [ ] Artifacts With Date in Date Range
- [ ] Artifacts Created in Date Range
- [ ] Pattern Search
- [ ] Builds for Dependency
- [ ] License Search
- [ ] Artifact Version Search
- [ ] Artifact Latest Version Search Based on Layout
- [ ] Artifact Latest Version Search Based on Properties
- [ ] Build Artifacts Search
- [x] List Docker Repositories
- [x] List Docker Tags

7.4 SECURITY

- [x] Get Users
- [x] Get User Details
- [ ] Get User Encrypted Password
- [x] Create or Replace User
- [x] Update User
- [x] Delete User
- [ ] Expire Password for a Single User
- [ ] Expire Password for Multiple Users
- [ ] Expire Password for All Users
- [ ] Unexpire Password for a Single User
- [ ] Change Password
- [ ] Get Password Expiration Policy
- [ ] Set Password Expiration Policy
- [ ] Configure User Lock Policy
- [ ] Retrieve User Lock Policy
- [x] Get Locked Out Users
- [x] Unlock Locked Out User
- [x] Unlock Locked Out Users
- [x] Unlock All Locked Out Users
- [x] Create API Key
- [x] Regenerate API Key
- [x] Get API Key
- [x] Revoke API Key
- [x] Revoke User API Key
- [ ] Revoke All API Keys
- [x] Get Groups
- [x] Get Group Details
- [x] Create or Replace Group
- [x] Update Group
- [x] Delete Group
- [x] Get Permission Targets
- [x] Get Permission Target Details
- [x] Create or Replace Permission Target
- [x] Delete Permission Target

- [x] Effective Item Permissions
- [ ] Security Configuration (Deprecated)
- [ ] Save Security Configuration (Deprecated)
- [ ] Activate Artifactory Key Encryption
- [ ] Deactivate Artifactory Key Encryption
- [ ] Set GPG Public Key
- [ ] Get GPG Public Key
- [ ] Set GPG Private Key
- [ ] Set GPG Pass Phrase
- [ ] Create Token
- [ ] Refresh Token
- [ ] Revoke Token
- [ ] Get Service ID
- [ ] Get Certificates
- [ ] Add Certificate
- [ ] Delete Certificate

7.5 REPOSITORIES

- [x] Get Repositories
- [x] Repository Configuration
- [x] Create Repository
- [x] Update Repository Configuration
- [x] Delete Repository
- [x] Calculate YUM Repository Metadata
- [x] Calculate NuGet Repository Metadata
- [x] Calculate Npm Repository Metadata
- [x] Calculate Maven Index
- [x] Calculate Maven Metadata
- [x] Calculate Debian Repository Metadata
- [x] Calculate Opkg Repository Metadata
- [x] Calculate Bower Index
- [x] Calculate Helm Chart Index

7.6 SYSTEM AND CONFIGURATION

- [x] System Info
- [x] System Health Ping
- [ ] Verify Connection
- [x] General Configuration
- [ ] Save General Configuration
- [ ] Update Custom URL Base
- [x] License Information
- [x] Install License
- [x] Version and Add-ons information
- [x] Get Reverse Proxy Configuration
- [ ] Update Reverse Proxy Configuration
- [x] Get Reverse Proxy Snippet

7.7 PLUGINS

- [ ] Execute Plugin Code
- [ ] Retrieve Plugin Info
- [ ] Retrieve Plugin Info Of A Certain Type
- [ ] Retrieve Build Staging Strategy
- [ ] Execute Build Promotion
- [ ] Reload Plugins

7.8 IMPORT AND EXPORT

- [ ] Import Repository Content
- [ ] Import System Settings Example
- [ ] Full System Import
- [ ] Export System Settings Example
- [ ] Export System

7.9 SUPPORT

- [ ] Create Bundle
- [ ] List Bundles
- [ ] Get Bundle

- [ ] Delete Bundle

CHAPTER 8

Changelog

8.1 1.4.9 (2020.07.06)

- Fixed settings override not working properly
- Dropped support for all the "SUPPORT" API methods (not working anymore)

8.2 1.4.8 (2019.02.10)

- Initial Open-source release

CHAPTER 9

rtpy package

9.1 rtpy.__init__.py

Exposed objects/functions to end user.

class rtpy.__init__.**Rtpy**(*settings*)

Main parent class.

Attributes are the classes are the methods categories.

Parameters **settings** (*dict*) – The user settings, mandaroty keys are “af_url” and “api_key” or “username” and “password”.

artifacts_and_storage

Category for multiple API methods

Type *rtpy.artifacts_and_storage.RtpyArtifactsAndStorage*

builds

Category for multiple API methods

Type *rtpy.builds.RtpyBuilds*

repositories

Category for multiple API methods

Type *rtpy.repositories.RtpyRepositories*

searches

Category for multiple API methods

Type *rtpy.searches.RtpySearches*

security

Category for multiple API methods

Type rtpy.security.RtpySecurity

support
Category for multiple API methods

Type rtpy.support.RtpySupport

system_and_configuration
Category for multiple API methods

Type *rtpy.system_and_configuration.RtpySystemAndConfiguration*

settings
Previously supplied settings at class instantiation

Type dict

rtpy.__init__.**json_to_dict**(*json_file_path*)
Convert a .json file to a Python dictionary.

Parameters **json_file_path** (*str*) – Path of the JSON file

Returns **dictionary** – The original JSON file as a Python dictionary

Return type dict

exception rtpy.__init__.**UserSettingsError**
Raised if some of the provided settings are incorrect or missing.

9.2 rtpy.artifacts_and_storage.py

Functions for the ARTIFACTS AND STORAGE REST API Methods category.

class rtpy.artifacts_and_storage.**RtpyArtifactsAndStorage**(*provided_settings*, *prefix*, *category*)
ARTIFACTS AND STORAGE methods category.

artifact_sync_download(*repo_key*, *artifact_path*, *options=None*, ***kwargs*)
Download an artifact.

With or without returning he actual content to the client.

When tracking the progress marks are printed (by default every 1024 bytes). This is extremely useful if you want to trigger downloads on a remote Artifactory server, for example to force eager cache population of large artifacts, but want to avoid the bandwidth consumption involved in transferring the artifacts to the triggering client. If no content parameter is specified the file content is downloaded to the client.

Parameters

- **repo_key** (*str*) – Key of the repository
- **artifact_path** (*str*) – Path of the artifact in the repository
- **options** (*str, optional*) – String of options
- ****kwargs** – Keyword arguments

copy_item(*src_repo_key*, *src_item_path*, *target_repo_key*, *target_item_path*, *options=None*, ***kwargs*)
Copy an artifact or a folder to the specified destination.

Supported by local repositories only.

Parameters

- **src_repo_key** (*str*) – Key of the source repository

- **src_item_path** (*str*) – Path of the item in the source repository
- **target_repo_key** (*str*) – Key of the target repository
- **target_item_path** (*str*) – Path of the item in the target repository
- **options** (*str, optional*) – String of options
- ****kwargs** – Keyword arguments

create_directory(*repo_key*, *directory_path*, ***kwargs*)
Create new directory at the specified destination.

Parameters

- **repo_key** (*str*) – Key of the repository
- **directory_path** (*str*) – Path of the directory in the repository
- ****kwargs** – Keyword arguments

delete_item(*repo_key*, *path_to_item*, ***kwargs*)
Delete a file or a folder from the specified destination.

Parameters

- **repo_key** (*str*) – Key of the repository
- **path_to_item** (*str*) – Path of the item in the repository
- ****kwargs** – Keyword arguments

delete_item_from_trash_can(*path_in_trashcan*, ***kwargs*)
Permanently delete an item from the trash can.

Parameters ****kwargs** – Keyword arguments

delete_item_properties(*repo_key*, *item_path*, *properties*, *options=None*, ***kwargs*)
Delete the specified properties from an item (file or folder).

When a folder is used property removal is recursive by default. Supported by local and local-cached repositories.

Parameters

- **repo_key** (*str*) – Key of the repository
- **item_path** (*str*) – The path of the item in the repository
- **properties** (*str*) – String of properties
- **options** (*str, optional*) – String of options
- ****kwargs** – Keyword arguments

deploy_artifact(*repo_key*, *local_artifact_path*, *target_artifact_path*, ***kwargs*)
Deploy an artifact to the specified destination.

Parameters

- **repo_key** (*str*) – Key of the repository
- **local_artifact_path** (*str*) – Local path of the artifact to upload
- **target_artifact_path** (*str*) – Target path of the artifact in the repository
- ****kwargs** – Keyword arguments

deploy_artifact_by_checksum(*repo_key*, *target_artifact_path*, *sha_type*, *sha_value*, ***kwargs*)

Deploy an artifact to the specified destination.

By checking if the artifact content already exists in Artifactory. If Artifactory already contains a user readable artifact with the same checksum the artifact content is copied over to the new location and return a response without requiring content transfer. Otherwise, a 404 error is returned to indicate that content upload is expected in order to deploy the artifact.

Parameters

- **repo_key** (*str*) – Key of the repository
- **target_artifact_path** (*str*) – Target path of the artifact in the repository
- **sha_type** (*str*) – Type of secure hash
- **sha_value** (*str*) – Value of the secure hash
- ****kwargs** – Keyword arguments

empty_trash_can(***kwargs*)

Empty the trash can permanently deleting all its current contents.

Parameters ****kwargs** – Keyword arguments

file_info(*repo_key*, *file_path*, ***kwargs*)

File Info.

For virtual use the virtual repository returns the resolved file. Supported by local, local-cached and virtual repositories.

Parameters

- **repo_key** (*str*) – Key of the repository
- **file_path** (*str*) – The path of the file in the repository
- ****kwargs** – Keyword arguments

file_list(*repo_key*, *folder_path*, *options=None*, ***kwargs*)

Get a flat (the default) or deep listing of the files and folders.

(not included by default) within a folder For deep listing you can specify an optional depth to limit the results. Optionally include a map of metadata timestamp values as part of the result

Parameters

- **repo_key** (*str*) – Key of the repository
- **folder_path** (*str*) – Path of the folder in the repository
- **options** (*str, optional*) – String of options
- ****kwargs** – Keyword arguments

file_statistics(*repo_key*, *item_path*, ***kwargs*)

Item statistics.

Record the number of times an item was downloaded, last download date and last downloader. Supported by local and local-cached repositories.

Parameters

- **repo_key** (*str*) – Key of the repository
- **item_path** (*str*) – The path of the item in the repository

- ****kwargs** – Keyword arguments

folder_info(*repo_key*, *folder_path*, ***kwargs*)
Folder Info.

For virtual use, the virtual repository returns the unified children. Supported by local, local-cached and virtual repositories.

Parameters

- **repo_key** (*str*) – Key of the repository
- **folder_path** (*str*) – The path of the folder in the repository
- ****kwargs** – Keyword arguments

get_background_tasks(***kwargs*)
Retrieve list of background tasks currently scheduled.

Or running in Artifactory In HA, the nodeId is added to each task. Task can be in one of few states: scheduled, running, stopped, canceled. Running task also shows the task start time.

Parameters ****kwargs** – Keyword arguments

get_storage_summary_info(***kwargs*)
Return storage summary information.

regarding binaries file store and repositories.

Parameters ****kwargs** – Keyword arguments

item_last_modified(*repo_key*, *item_path*, ***kwargs*)
Retrieve the last modified item at the given path.

If the given path is a folder, the latest last modified item is searched for recursively. Supported by local and local-cached repositories.

Parameters

- **repo_key** (*str*) – Key of the repository
- **item_path** (*str*) – The path of the item in the repository
- ****kwargs** – Keyword arguments

item_properties(*repo_key*, *item_path*, *properties=None*, ***kwargs*)
Item Properties. Optionally return only the properties requested.

Parameters

- **repo_key** (*str*) – Key of the repository
- **item_path** (*str*) – The path of the item in the repository
- **properties** (*str*) – String of properties
- ****kwargs** – Keyword arguments

move_item(*src_repo_key*, *src_item_path*, *target_repo_key*, *target_item_path*, *options=None*, ***kwargs*)
Move an artifact or a folder to the specified destination.

Supported by local repositories only.

Parameters

- **src_repo_key** (*str*) – Key of the source repository
- **src_item_path** (*str*) – Path of the item in the source repository

- **target_repo_key** (*str*) – Key of the target repository
- **target_item_path** (*str*) – Path of the item in the target repository
- **options** (*str, optional*) – String of options
- ****kwargs** – Keyword arguments

optimize_system_storage(***kwargs*)
Raise a flag to invoke balancing between redundant storage units.

Of a sharded filestore following the next garbage collection.

restore_item_from_trash_can(*path_in_trashcan*, *target_path*, ***kwargs*)
Restore an item from the trash can.

Parameters

- **path_in_trashcan** (*str*) – Path of the item in the trashcan (repo_name/folder/file)
- **target_path** (*str*) – Where to restore the item (repo_name/folder/file)

retrieve_artifact(*repo_key*, *artifact_path*, ***kwargs*)
Retrieve an artifact from the specified destination.

Parameters

- **repo_key** (*str*) – Key of the repository
- **artifact_path** (*str*) – Path of the artifact in the repository
- ****kwargs** – Keyword arguments

retrieve_folder_or_repository_archive(*repo_key*, *path*, *archive_type*, *include_checksums=False*, ***kwargs*)
Retrieve an archive file (supports zip/tar/tar.gz/tgz).

containing all the artifacts that reside under the specified path (folder or repository root). Requires Enable Folder Download to be set.

Parameters

- **repo_key** (*str*) – Key of the repository
- **path** (*str*) – Path of the folder in the repository
- **archive_type** (*str*) – Type of archive
- **include_checksums** (*bool, optional*) – True to include checksums, False by default
- ****kwargs** – Keyword arguments

set_item_properties(*repo_key*, *item_path*, *properties*, *options=None*, ***kwargs*)
Attach properties to an item (file or folder).

When a folder is used property attachment is recursive by default. Supported by local and local-cached repositories

Parameters

- **repo_key** (*str*) – Key of the repository
- **item_path** (*str*) – The path of the item in the repository
- **properties** (*str*) – String of properties
- **options** (*str, optional*) – String of options

- ****kwargs** – Keyword arguments

set_item_sha256_checksum(*params*, ***kwargs*)
Calculate an artifact's SHA256 checksum and attaches it as a property.

(with key "sha256"). If the artifact is a folder, then recursively calculates the SHA256 of each item in the folder and attaches the property to each item.

Parameters

- **params** (*dict*) – Dictionary comprised of {"repo_key": str, "path": str}
- ****kwargs** – Keyword arguments

trace_artifact_retrieval(*repo_key*, *item_path*, ***kwargs*)
Simulate an artifact retrieval request from the specified.

location and returns verbose output about the resolution process.

Parameters

- **repo_key** (*str*) – Key of the repository
- **item_path** (*str*) – The path of the item in the repository
- ****kwargs** – Keyword arguments

9.3 rtpy.builds.py

Functions for the BUILDS REST API Methods category.

class rtpy.builds.**RtpyBuilds**(*provided_settings*, *prefix*, *category*)
BUILDS methods category.

all_builds(***kwargs*)
Provide information on all builds.

Parameters ****kwargs** – Keyword arguments

9.4 rtpy.import_and_export.py

Functions for the IMPORT AND EXPORT REST API Methods category.

9.5 rtpy.plugins.py

Functions for the PLUGINS REST API Methods category.

9.6 rtpy.repositories.py

Functions for the REPOSITORIES REST API Methods category.

class rtpy.repositories.**RtpyRepositories**(*provided_settings*, *prefix*, *category*)
REPOSITORIES methods category.

calculate_bower_index(*repo_key*, ***kwargs*)
Recalculate the index for a Bower repository.

Parameters

- **repo_key** (*str*) – Key of the repository
- ****kwargs** – Keyword arguments

calculate_conda_repository_metadata(*repo_key*, *options=None*, ***kwargs*)
Calculate/recalculate the Conda packages and release metadata for a repository.

The calculation can be synchronous (the default) or asynchronous. Please refer to Conda Repositories for more details. Supported for local repositories only

Parameters

- **repo_key** (*str*) – Key of the repository
- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

calculate_cran_repository_metadata(*repo_key*, *options=None*, ***kwargs*)
Calculates/recalculates the Packages and Release metadata for a repository.

Based on the CRAN packages in it. The calculation can be synchronous (the default) or asynchronous. Please refer to CRAN Repositories for more details. Supported by local repositories only. From version 6.1, by default, the recalculation process also writes several entries from the CRAN package's metadata as properties on all of the artifacts (based on the control file's content).

Parameters

- **repo_key** (*str*) – Key of the repository
- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

calculate_debian_repository_metadata(*repo_key*, *x_gpg_passphrase=None*, *options=None*, ***kwargs*)
Calculate/recalculate the Packages and Release metadata.

for this repository, based on the Debian packages in it. Calculation can be synchronous (the default) or asynchronous. Please refer to Debian Repositories for more details. Supported by local repositories only. From version 4.4, by default, the recalculation process also writes several entries from the Debian package's metadata as properties on all of the artifacts (based on the control file's content). This operation may not always be required (for example, if the Debian files are intact and were not modified, only the index needs to be recalculated. The operation is resource intensive and can be disabled by passing the ?writeProps=0 query param. From version 5.7, the target repository can be a virtual repository.

Parameters

- **repo_key** (*str*) – Key of the repository
- **x_gpg_passphrase** (*str*) – Passphrase
- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

calculate_helm_chart_index(*repo_key*, ***kwargs*)
Calculate Helm chart index on the specified path.

(local repositories only).

Parameters

- **repo_key** (*str*) – Key of the repository
- ****kwargs** – Keyword arguments

calculate_maven_index(*options*, ***kwargs*)
Calculates/caches a Maven index for the specified repositories.

For a virtual repository specify all underlying repositories that you want the aggregated index to include. Calculation can be forced, which for remote repositories will cause downloading of a remote index even if a locally ached index has not yet expired; and index recalculation based on the cache on any failure to download the remote index, including communication errors (the default behavior is to only use the cache when a remote index cannot be found and returns a 404). Forcing has no effect on local repositories index calculation.

Parameters

- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

calculate_maven_metadata(*repo_key*, *folder_path*, *options=None*, ***kwargs*)
Calculate Maven metadata on the specified path.

(local repositories only).

Parameters

- **repo_key** (*str*) – Key of the repository
- **folder_path** (*str*) – Path of the folder in the repository
- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

calculate_npm_repository_metadata(*repo_key*, ***kwargs*)
Recalculate the npm search index for this repository (local/virtual).

Please see the Npm integration documentation for more details. Supported by local and virtual repositories.

Parameters

- **repo_key** (*str*) – Key of the repository
- ****kwargs** – Keyword arguments

calculate_nuget_repository_metadata(*repo_key*, ***kwargs*)
Recalculate all the NuGet packages for a repository.

(local/cache/virtual), and re-annotate the NuGet properties for each NuGet package according to it's internal nuspec file. Please see the NuGet integration documentation for more details. Supported by local, local-cache, remote and virtual repositories.

Parameters

- **repo_key** (*str*) – Key of the repository
- ****kwargs** – Keyword arguments

calculate_opkg_repository_metadata(*repo_key*, *x_gpg_passphrase=None*, *options=None*, ***kwargs*)
Calculate/recalculate the Packages and Release metadata fora repository.

Based on the ipk packages in it (in each feed location). Calculation can be synchronous (the default) or asynchronous. Please refer to Opkg Repositories for more details. Supported by local repositories only. By default, the recalculation process also writes several entries from the ipk package's metadata as properties on all of the artifacts (based on the control file's content). This operation may not always be required (for

example, if the ipk files are intact and were not modified, only the index needs to be recalculated. The operation is resource intensive and can be disabled by passing the ?writeProps=0 query param.

Parameters

- **repo_key** (*str*) – Key of the repository
- **x_gpg_passphrase** (*str*) – Passphrase
- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

calculate_yum_repository_metadata(*repo_key*, *x_gpg_passphrase=None*, *options=None*, ***kwargs*)

Calculate/recalculate the YUM metdata for a repository.

Based on the RPM package currently hosted in the repository. Supported by local and virtual repositories only. Calculation can be synchronous (the default) or asynchronous. For Virtual repositories, calculates the merged metadata from all aggregated repositories on the specified path. The path parameter must be passed for virtual calculation.

Parameters

- **repo_key** (*str*) – Key of the repository
- **x_gpg_passphrase** (*str*) – Passphrase
- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

create_repository(*params*, ***kwargs*)

Create a new repository in Artifactory with the provided configuration.

Supported by local, remote and virtual repositories. A position may be specified using the pos parameter. If the map size is shorter than pos the repository is the last one (the default behavior).

Parameters

- **params** (*dict*) – Parameters of the repository
- ****kwargs** – Keyword arguments

delete_repository(*repo_key*, ***kwargs*)

Remove a repository.

Configuration together with the whole repository content. Supported by local, remote and virtual repositories

Parameters

- **repo_key** (*str*) – Key of the repository
- ****kwargs** – Keyword arguments

get_repositories(*options=None*, ***kwargs*)

Return a list of minimal repository details.

For all repositories of the specified type.

Parameters

- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

repository_configuration(*repo_key*, ***kwargs*)
Retrieve the current configuration of a repository.

Supported by local, remote and virtual repositories.

Parameters

- **repo_key** (*str*) – Key of the repository
- ****kwargs** – Keyword arguments

update_repository_configuration(*params*, ***kwargs*)
Update an exiting repository configuration in Artifactory.

With the provided configuration elements. Supported by local, remote and virtual repositories.

Parameters

- **params** (*dict*) – Parameters of the repository
- ****kwargs** – Keyword arguments

9.7 rtpy.rtpy.py

Rtpy class definition with it's attributes which is exposed to the end user.

class rtpy.rtpy.**Rtpy**(*settings*)
Main parent class.

Attributes are the classes are the methods categories.

Parameters **settings** (*dict*) – The user settings, mandaroty keys are "af_url" and "api_key" or "username" and "password".

artifacts_and_storage
Category for multiple API methods

Type *rtpy.artifacts_and_storage.RtpyArtifactsAndStorage*

builds
Category for multiple API methods

Type *rtpy.builds.RtpyBuilds*

repositories
Category for multiple API methods

Type *rtpy.repositories.RtpyRepositories*

searches
Category for multiple API methods

Type *rtpy.searches.RtpySearches*

security
Category for multiple API methods

Type rtpy.security.RtpySecurity

support
Category for multiple API methods

Type rtpy.support.RtpySupport

system_and_configuration
Category for multiple API methods

Type *rtpy.system_and_configuration.RtpySystemAndConfiguration*

settings
Previously supplied settings at class instantiation

Type dict

9.8 rtpy.searches.py

Functions for the SEARCHES REST API Methods category.

class rtpy.searches.**RtpySearches**(*provided_settings*, *prefix*, *category*)
SEARCHES methods category.

artifactory_query_language(*query*, ***kwargs*)
Search items using the Artifactory Query Language (AQL).

Parameters

- **query** (*str*) – The AQL string
- ****kwargs** – Keyword arguments

list_docker_repositories(*repo_key*, *options=None*, ***kwargs*)
List all Docker repositories (the registry's _catalog).

(Hosted in an Artifactory Docker repository).

Parameters

- **repo_key** (*str*) – Key of the repository
- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

list_docker_tags(*repo_key*, *image_path*, *options=None*, ***kwargs*)
List all tags of the specified Artifactory Docker repository.

Parameters

- **repo_key** (*str*) – Key of the repository
- **image_path** (*str*) – Path of the image in the repository
- **options** (*str*) – String of options
- ****kwargs** – Keyword arguments

9.9 rtpy.support.py

9.10 rtpy.system_and_configuration.py

Functions for the SYSTEM AND CONFIGURATION REST API Methods category.

class rtpy.system_and_configuration.**RtpySystemAndConfiguration**(*provided_settings*, *prefix*, *category*)

SYSTEM AND CONFIGURATION methods category.

general_configuration(***kwargs*)
Get the general configuration (artifactory.config.xml).

Parameters ****kwargs** – Keyword arguments

get_reverse_proxy_configuration(***kwargs*)
Retrieve the reverse proxy configuration.

Parameters ****kwargs** – Keyword arguments

get_reverse_proxy_snippet(***kwargs*)
Get the reverse proxy configuration snippet in text format.

Parameters ****kwargs** – Keyword arguments

install_license(*params*, ***kwargs*)
Install new license key or change the current one.

Parameters

- **params** (*str*) – Settings of the license
- ****kwargs** – Keyword arguments

license_information(***kwargs*)
Retrieve information about the currently installed license.

Parameters ****kwargs** – Keyword arguments

save_general_configuration(*xml_file_path*, ***kwargs*)
Save the general configuration (artifactory.config.xml).

Parameters

- **xml_file_path** (*str*) – Path of the xml file to POST
- ****kwargs** – Keyword arguments

system_health_ping(***kwargs*)
Get a simple status response about the state of Artifactory.

Parameters ****kwargs** – Keyword arguments

system_info(***kwargs*)
Get general system information.

Parameters ****kwargs** – Keyword arguments

version_and_addons_information(***kwargs*)
Retrieve information about versions and addons.

(the current Artifactory version, revision, and currently installed Add-ons).

Parameters ****kwargs** – Keyword arguments

9.11 rtpy.tools.py

Functions and classes used by the main categories of methods.

class rtpy.tools.**RtpyBase**(*provided_settings*, *prefix*, *category*)
Rtpy Base class.

_prefix
API endpoint used for a given API method category ("search/"...)

Type str

_category
Major API method category (Searches, Repositories...)

Type str

_user_settings
Complete user settings, default values are changed at class instantiation

Type dict

exception AfApiError(*error_details*)
Raised when encountering a standard Artifactory REST API error JSON.

api_method
Name of the specific method (category and name)
Type str

url
Full URL used for API call
Type str

verb
HTTP verb used for the API call ("GET", "POST"...)
Type str

status_code
HTTP status code for the API call
Type int

message
Error message given by the Artifactory REST API
Type str

print_message
Well formatted multiline message with all the attributes
Type str

exception MalformedAfApiError(**args*, ***kwargs*)
Raised when encountering a malformed Artifactory REST API error JSON.

exception RtpyError
Raised if arguments are wrong or incomplete for a method.

This error is used as a preemptive measure in some methods to check arguments.

exception rtpy.tools.**UserSettingsError**
Raised if some of the provided settings are incorrect or missing.

rtpy.tools.**json_to_dict**(*json_file_path*)
Convert a .json file to a Python dictionary.

Parameters **json_file_path** (*str*) – Path of the JSON file

Returns **dictionary** – The original JSON file as a Python dictionary

Return type dict

9.12 rtpy.xray.py

Functions for the XRAY REST API Methods category.

Python Module Index

r

Index

Symbols

A

B

C

D

E

F

G

I

J

L

M

O

P

R

S

T

U

V