
Zowe Client Python SDK

Release 1.0.0.dev15

Contributors to the Zowe Project

Apr 12, 2024

CONTENTS:

1	Getting started	3
1.1	Installation	3
1.2	Basic usage	3
2	About the project	5
3	Zowe SDK packages	7
3.1	Core	7
3.2	z/OS Console	16
3.3	z/OS Files	17
3.4	z/OS Jobs	23
3.5	z/OS TSO	27
3.6	z/OSMF	29
4	Classes	31
4.1	zowe.secrets_for_zowe_sdk	31
5	Contributing	33
5.1	Git branching model	33
5.2	Pull requests	33
5.3	Running Tests	34
5.4	Building Docs	34
5.5	Code standards	34
	Index	35

The Zowe Client Python SDK, is a set of Python packages designed to allow programmatic interactions with z/OS REST API interfaces.

Python developers can leverage the Zowe SDK in order to create powerful scripts/applications that can interact with z/OS components.

GETTING STARTED

1.1 Installation

When installing the Zowe Client Python SDK you have two options.

- Install all the Zowe packages
- Install a single sub-package

The choice depends on your intentions. If you chose to install all Zowe SDK packages this means that you will install everything under the *zowe* namespace in PyPi.

Alternatively you can chose to install only a single subpackage for a smaller installation.

To install all Zowe SDK packages using pip:

```
pip install -U --pre zowe
```

To install only a subpackage using pip:

```
pip install -U --pre zowe.<subpackage>_for_zowe_sdk
```

To see all available sub-packages check the [Zowe SDK packages](#) section.

1.2 Basic usage

After you install the package in your project, import the class for the required sub-package (i.e *Console* class for z/OS Console commands). Create a dictionary to handle communication with the plug-in:

```
from zowe.zos_console_for_zowe_sdk import Console
profile = {
    "host": "<host address>",
    "port": 443, # Include the port if different from the default (443)
    "user": "<user>",
    "password": "<password>",
}

my_console = Console(profile)
```

Alternatively you can use an existing Zowe CLI profile instead:

```
from zowe.zos_console_for_zowe_sdk import Console
from zowe.core_for_zowe_sdk import ProfileManager

# Load the profile using ProfileManager
profile = ProfileManager().load(profile_name="<profile name>")

my_console = Console(profile)
```


ABOUT THE PROJECT

The Zowe Client Python SDK, is a set of Python packages designed to allow programmatic interactions with z/OS REST API interfaces.

Python developers can leverage the Zowe SDK in order to create powerful scripts/applications that can interact with z/OS components.

ZOWE SDK PACKAGES

The Zowe Client Python SDK is divided into multiple packages. Each one responsible for a single aspect of the SDK.

3.1 Core

Contains the core functionalities for the Zowe Client Python SDK.

3.1.1 Installing

To install this package using pip issue the following command:

```
pip install zowe.zowe_core_for_zowe_sdk
```

3.1.2 Reference

zowe.core_for_zowe_sdk

ApiConnection

class `core.zowe.core_for_zowe_sdk.ApiConnection`(*host_url, user, password, ssl_verification=True*)

Class used to represent a connection with a REST API.

host_url

The base url of the rest api host

Type

str

user

The user of the rest api

Type

str

password

The password for the user

Type

str

ssl_verification

Type
bool

CredentialManager

class core.zowe.core_for_zowe_sdk.CredentialManager

static load_secure_props() → None

load secure_props stored for the given config file :returns: * None

- if keyring is not initialized, set empty value

static save_secure_props() → None

Set secure_props for the given config file :rtype: None

ProfileManager

class core.zowe.core_for_zowe_sdk.ProfileManager(appname: str = 'zowe', show_warnings: bool = True)

Profile Manager contains the logic to merge the different properties of profiles (from the Project Config and the Project User Config as well as the Global Config and Global User Config). This class handles all the exceptions raised in the Config File to provide a smooth user experience.

property config_appname: str

Returns the app name

property config_dir: str | None

Returns the folder path to where the Zowe z/OSMF Team Project Config files are located.

property config_filename: str

Return the filename for Zowe z/OSMF Team Project Config

property config_filepath: str | None

Get the full Zowe z/OSMF Team Project Config filepath

static get_env(cfg: ConfigFile, cwd=None) → dict

Maps the env variables to the profile properties

Returns

Containing profile properties from env variables (prop: value)

Return type

Dictionary

get_highest_priority_layer(json_path: str) → ConfigFile | None

Get the highest priority layer (configuration file) based on the given profile name

Parameters:

profile_name (str): The name of the profile to look for in the layers.

Returns:

Optional[ConfigFile]: The highest priority layer (configuration file) that contains the specified profile,
or None if the profile is not found in any layer.

static get_profile(cfg: [ConfigFile](#), profile_name: str | None, profile_type: str | None, validate_schema: bool | None = True) → [Profile](#)

Get just the profile from the config file (overridden with base props in the config file)

Returns

NamedTuple (data, name, secure_props_not_found)

Return type

[Profile](#)

load(profile_name: str | None = None, profile_type: str | None = None, check_missing_props: bool = True, validate_schema: bool | None = True, override_with_env: bool | None = False) → dict

Load connection details from a team config profile. :returns: * *dictionary* – Object containing connection details

- We will load properties from config files in the following order, from
- highest to lowest priority
- 1. Project User Config (./zowe.config.user.json)
- 2. Project Config (./zowe.config.json)
- 3. Global User Config (~/.zowe.config.user.json)
- 4. Global Config (~/.zowe.config.json)
- If *profile_type* is not base, then we will load properties from both
- *profile_type* and base profiles and merge them together.

save() → None

Save the layers (configuration files) to disk.

set_profile(profile_path: str, profile_data: dict) → None

Set a profile in the highest priority layer (configuration file) based on the given profile name

Parameters:

profile_path (str): The path of the profile to be set. eg: profiles.zosmf profile_data (dict): The data of the profile to set.

set_property(json_path, value, secure=None) → None

Set a property in the profile, storing it securely if necessary.

Parameters:

json_path (str): The JSON path of the property to set. value (str): The value to be set for the property. secure (bool): If True, the property will be stored securely. Default is None.

property_user_config_dir: str | None

Returns the folder path to where the Zowe z/OSMF User Project Config files are located.

RequestHandler

class `core.zowe.core_for_zowe_sdk.RequestHandler(session_arguments)`

Class used to handle HTTP/HTTPS requests.

session_arguments

Zowe SDK session arguments

Type

dict

valid_methods

List of supported request methods

Type

list

perform_request(*method, request_arguments, expected_code=[200]*)

Execute an HTTP/HTTPS requests from given arguments and return validated response (JSON).

Parameters

- **method** (*str*) – The request method that should be used
- **request_arguments** (*dict*) – The dictionary containing the required arguments for the execution of the request
- **expected_code** (*int*) – The list containing the acceptable response codes (default is [200])

Returns

normalized_response – normalized request response in json (dictionary)

Return type

json

perform_streamed_request(*method, request_arguments, expected_code=[200]*)

Execute a streamed HTTP/HTTPS requests from given arguments and return a raw response.

Parameters

- **method** (*str*) – The request method that should be used
- **request_arguments** (*dict*) – The dictionary containing the required arguments for the execution of the request
- **expected_code** (*int*) – The list containing the acceptable response codes (default is [200])

Return type

A raw response data

SdkApi

```
class core.zowe.core_for_zowe_sdk.SdkApi(profile, default_url)
```

Abstract class used to represent the base SDK API.

ZosmfProfile

```
class core.zowe.core_for_zowe_sdk.ZosmfProfile(profile_name)
```

Class used to represent a Zowe z/OSMF profile.

Description

This class is only used when there is already a Zowe z/OSMF profile created and the user opted to use the profile instead of passing the credentials directly in the object constructor.

profile_name

Zowe z/OSMF profile name

Type

str

load()

Load z/OSMF connection details from a z/OSMF profile.

Returns

z/OSMF connection object

Return type

zosmf_connection

property profiles_dir

Return the os path for the Zowe z/OSMF profiles.

Config File classes

ConfigFile

```
class core.zowe.core_for_zowe_sdk.config_file.ConfigFile(type: str, name: str, _location: str | None
                                                         = None, profiles: dict | None = None,
                                                         defaults: dict | None = None,
                                                         schema_property: dict | None = None,
                                                         secure_props: dict | None = None, jsonc:
                                                         dict | None = None,
                                                         _missing_secure_props: list =
                                                         <factory>)
```

Class used to represent a single config file.

Mainly it will have the following details : 1. Type (“User Config” or “Team Config”)

User Configs override Team Configs. User Configs are used to have personalised config details that the user don’t want to have in the Team Config.

2. Directory in which the file is located.

3. Name (excluding .config.json or .config.user.json)
4. Contents of the file. 4.1 Profiles 4.2 Defaults 4.3 Schema Property 5. Secure Properties associated with the file.

autodiscover_config_dir() → None

Autodiscover Zowe z/OSMF Team Config files by going up the path from current working directory :returns: * None

- Sets path if it finds the config directory,
- Else, it raises an Exception

find_profile(path: str, profiles: dict)

Find a profile at a specified location from within a set of nested profiles :returns: The profile object that was found, or None if not found :rtype: dictionary

get_profile(profile_name: str | None = None, profile_type: str | None = None, validate_schema: bool | None = True) → Profile

Load given profile including secure properties and excluding values from base profile :returns: Returns a namedtuple called Profile :rtype: Profile

get_profile_name_from_path(path: str) → str

Get the name of the profile from the given path.

get_profile_path_from_name(short_path: str) → str

Get the path of the profile from the given name.

get_profilename_from_profiletype(profile_type: str) → str

Returns profilename from given profiletype as defined in the team config profile :returns: * str

- Return the exact profilename of the profile to load from the mentioned type.
- First tries to look into the defaults, if not found,
- then it tries to iterate through the profiles

init_from_file(validate_schema: bool | None = True) → None

Initializes the class variable after setting filepath (or if not set, autodiscover the file)

load_profile_properties(profile_name: str) → dict

Load profile properties given profile_name including secure properties :returns: * dictionary – Object containing profile properties

- Load exact profile properties (without prepopulated fields from base profile)
- from the profile dict and populate fields from the secure credentials storage

save(update_secure_props=True)

Save the config file to disk. and secure props to vault parameters:

secure_props (bool): If True, the secure properties will be stored in the vault. Default is True.

Returns:

None

schema_list(cwd=None) → list

Loads the schema properties in a sorted order according to the priority

Returns

Returns the profile properties from schema (prop: value)

Return type

Dictionary

set_profile(*profile_path*: str, *profile_data*: dict) → None

Set a profile in the config file.

Parameters:*profile_path* (str): The path of the profile to be set. eg: profiles.zosmf *profile_data* (dict): The data to be set for the profile.**set_property**(*json_path*, *value*, *secure=None*) → None

Set a property in the profile, storing it securely if necessary.

Parameters:*json_path* (str): The JSON path of the property to set. *value* (str): The value to be set for the property. *profile_name* (str): The name of the profile to set the property in. *secure* (bool): If True, the property will be stored securely. Default is None.**validate_schema**() → None

Get the \$schema_property from the config and load the schema

Return type

file_path to the \$schema property

Profile**class** core.zowe.core_for_zowe_sdk.config_file.Profile(*data*, *name*, *missing_secure_props*)**data:** dict

Alias for field number 0

missing_secure_props: list

Alias for field number 2

name: str

Alias for field number 1

Custom Warnings classes**ConfigNotFoundWarning****class** core.zowe.core_for_zowe_sdk.custom_warnings.ConfigNotFoundWarning(*message*)**ProfileNotFoundWarning****class** core.zowe.core_for_zowe_sdk.custom_warnings.ProfileNotFoundWarning(*message*)

ProfileParsingWarning

```
class core.zowe.core_for_zowe_sdk.custom_warnings.ProfileParsingWarning(message)
```

SecurePropsNotFoundWarning

```
class core.zowe.core_for_zowe_sdk.custom_warnings.SecurePropsNotFoundWarning(message)
```

Exceptions classes

FileNotFound

```
class core.zowe.core_for_zowe_sdk.exceptions.FileNotFound(input_path)
```

Class used to represent a file not found exception.

InvalidRequestMethod

```
class core.zowe.core_for_zowe_sdk.exceptions.InvalidRequestMethod(input_method)
```

Class used to represent an invalid request method exception.

MissingConnectionArgs

```
class core.zowe.core_for_zowe_sdk.exceptions.MissingConnectionArgs
```

Class used to represent a missing connection argument exception.

ProfileNotFound

```
class core.zowe.core_for_zowe_sdk.exceptions.ProfileNotFound(profile_name: str = 'unknown',  
                                                             error_msg: str = 'error')
```

Class used to represent a profile load failure exception.

RequestFailed

```
class core.zowe.core_for_zowe_sdk.exceptions.RequestFailed(status_code, request_output)
```

Class used to represent a request failure exception.

SecureProfileLoadFailed

```
class core.zowe.core_for_zowe_sdk.exceptions.SecureProfileLoadFailed(profile_name: str =  
                                                                    'unknown', error_msg: str  
                                                                    = 'error')
```

Class used to represent a secure profile load failure exception.

SecureValuesNotFound

```
class core.zowe.core_for_zowe_sdk.exceptions.SecureValuesNotFound(values: set)
```

Class used to represent a profile load failure exception.

UnexpectedStatus

```
class core.zowe.core_for_zowe_sdk.exceptions.UnexpectedStatus(expected, received, request_output)
```

Class used to represent an unexpected request response status exception.

UnsupportedAuthType

```
class core.zowe.core_for_zowe_sdk.exceptions.UnsupportedAuthType(auth_type: str)
```

Class used to represent an unsupported authentication type exception.

Session classes

ISession

```
class core.zowe.core_for_zowe_sdk.session.ISession(host: str, port: int = 443, rejectUnauthorized:  
                                                    bool = True, user: str | None = None, password:  
                                                    str | None = None, protocol: str = 'https',  
                                                    basePath: str | None = None, type: str | None =  
                                                    None, tokenType: str | None = None, tokenValue:  
                                                    str | None = None)
```

Class to represent session parameters

Session

```
class core.zowe.core_for_zowe_sdk.session.Session(props: dict)
```

Class used to set connection details received from a ProfileManager or manually set by passing and ISession object

3.2 z/OS Console

Contains the z/OSMF Console REST API functionalities.

3.2.1 Installing

```
pip install zowe.zos_console_for_zowe_sdk
```

3.2.2 Reference

`zowe.zos_console_for_zowe_sdk`

Console

class `zos_console.zowe.zos_console_for_zowe_sdk.Console(connection)`

Class used to represent the base z/OSMF Console API.

...

connection

Connection object

get_response(*response_key*, *console=None*)

Collect outstanding synchronous z/OS Console response messages.

Parameters

- **response_key** – The command response key from the Issue Command request.
- **console** – The console that should be used to get the command response.

Returns

A JSON containing the response to the command

Return type

json

issue_command(*command*, *console=None*)

Issues a command on z/OS Console.

Parameters

- **command** – The z/OS command to be executed
- **console** – The console that should be used to execute the command (default is None)

Returns

A JSON containing the response from the console command

Return type

json

3.3 z/OS Files

Contains the z/OSMF Files REST API functionalities.

3.3.1 Installing

```
pip install zowe.zos_files_for_zowe_sdk
```

3.3.2 Reference

zowe.zos_files_for_zowe_sdk

FileType

Files

class `zos_files.zowe.zos_files_for_zowe_sdk.Files(connection)`

Class used to represent the base z/OSMF Files API.

...

connection

connection object

copy_dataset_or_member(*from_dataset_name, to_dataset_name, from_member_name=None, volser=None, alias=None, to_member_name=None, enq=None, replace=False*)

Copy a dataset or member to another dataset or member. :param from_dataset_name: Name of the dataset to copy from :type from_dataset_name: str :param to_dataset_name: Name of the dataset to copy to :type to_dataset_name: str :param from_member_name: Name of the member to copy from :type from_member_name: str :param volser: Volume serial number of the dataset to copy from :type volser: str :param alias: Alias of the dataset to copy from :type alias: bool :param to_member_name: Name of the member to copy to :type to_member_name: str :param enq: Enqueue type for the dataset to copy from :type enq: str :param replace: If true, members in the target data set are replaced. :type replace: bool

Returns

A JSON containing the result of the operation

Return type

json

copy_uss_to_dataset(*from_filename, to_dataset_name, to_member_name=None, type=FileType.TEXT, replace=False*)

Copy a USS file to dataset.

Parameters

- **from_filename** (*str*) – Name of the file to copy from.
- **to_dataset_name** (*str*) – Name of the dataset to copy to.
- **to_member_name** (*str*) – Name of the member to copy to.
- **type** (*FileType, optional*) – Type of the file to copy from. Default is `FileType.TEXT`.
- **replace** (*bool, optional*) – If true, members in the target dataset are replaced.

Returns

A JSON containing the result of the operation.

Return type

json

create_data_set(*dataset_name*, *options*={})

Create a sequential or partitioned dataset. :param dataset_name:

Return type

json

create_default_data_set(*dataset_name*: str, *default_type*: str)

Create a dataset with default options set. Default options depend on the requested type.

Parameters

- **dataset_name** (str) –
- **default_type** (str) – “partitioned”, “sequential”, “classic”, “c” or “binary”

Return type

json - A JSON containing the result of the operation

create_uss(*file_path*, *type*, *mode*=None)

Add a file or directory :param file_path of the file to add: :param type = “file” or “dir”: :param mode Ex: :type mode Ex: - rwxr-xr-x

create_zFS_file_system(*file_system_name*, *options*={})

Create a z/OS UNIX zFS Filesystem.

Parameter

file_system_name: str - the name for the file system

rtype

json - A JSON containing the result of the operation

delete_data_set(*dataset_name*, *volume*=None, *member_name*=None)

Deletes a sequential or partitioned data.

delete_migrated_data_set(*dataset_name*: str, *purge*=False, *wait*=False)

Deletes migrated data set.

Parameters

- **dataset_name** (str) – Name of the data set
- **purge** (bool) – If true, the function uses the PURGE=YES on ARCHDEL request, otherwise it uses the PURGE=NO.
- **wait** (bool) – If true, the function waits for completion of the request, otherwise the request is queued.

Return type

json - A JSON containing the result of the operation

delete_uss(*filepath_name*, *recursive*=False)

Delete a file or directory

Parameters

- **deleted** (*filepath of the file to be*) –
- **recursive** – If specified as True, all the files and sub-directories will be deleted.

Returns

HTTP Response for No Content

Return type

204

delete_zFS_file_system(*file_system_name*)

Deletes a zFS Filesystem

download_binary_dsn(*dataset_name, output_file, with_prefixes=False*)

Retrieve the contents of a binary dataset and saves it to a given file.

Parameters

- **dataset_name** (*str - Name of the dataset to download*) –
- **output_file** (*str - Name of the local file to create*) –
- **with_prefixes** (*boolean - If true, include a four big endian bytes record length prefix.*) – The default is False

download_dsn(*dataset_name, output_file*)

Retrieve the contents of a dataset and saves it to a given file.

download_uss(*file_path, output_file, binary=False*)

Retrieve the contents of a USS file and saves it to a local file.

get_dsn_binary_content(*dataset_name, with_prefixes=False*)

Retrieve the contents of a given dataset as a binary bytes object.

Parameters

- **dataset_name** (*str - Name of the dataset to retrieve*) –
- **with_prefixes** (*boolean - if True include a 4 byte big endian record len prefix*) – default: False

Returns

A response object from the requests library

Return type

response

get_dsn_binary_content_streamed(*dataset_name, with_prefixes=False*)

Retrieve the contents of a given dataset as a binary bytes object streamed.

Parameters

- **dataset_name** (*str - Name of the dataset to retrieve*) –
- **with_prefixes** (*boolean - if True include a 4 byte big endian record len prefix*) – default: False

Returns

A response object from the requests library

Return type

response

get_dsn_content(*dataset_name*)

Retrieve the contents of a given dataset.

Returns

A JSON with the contents of a given dataset

Return type

json

get_dsn_content_streamed(*dataset_name*)

Retrieve the contents of a given dataset streamed.

Returns

A response object from the requests library

Return type

response

get_file_content(*filepath_name*)

Retrieve the content of a filename. The complete path must be specified.

Returns

A JSON with the contents of the specified USS file

Return type

json

get_file_content_streamed(*file_path*, *binary=False*)

Retrieve the contents of a given USS file streamed.

Returns

A response object from the requests library

Return type

response

list_dsn(*name_pattern*, *return_attributes=False*)

Retrieve a list of datasets based on a given pattern.

Parameters

- **name_pattern** (*str*) – The pattern to match dataset names.
- **return_attributes** (*bool*, *optional*) – Whether to return dataset attributes along with the names. Defaults to False.

Returns

- *list of dict*
- *A JSON with a list of dataset names (and attributes if specified) matching the given pattern.*

list_dsn_members(*dataset_name*, *member_pattern=None*, *member_start=None*, *limit=1000*, *attributes='member'*)

Retrieve the list of members on a given PDS/PDSE.

Returns

A JSON with a list of members from a given PDS/PDSE

Return type

json

list_files(path)

Retrieve a list of USS files based on a given pattern.

Returns

A JSON with a list of dataset names matching the given pattern

Return type

json

list_unix_file_systems(file_path_name=None, file_system_name=None)

list all mounted filesystems, or the specific filesystem mounted at a given path, or the filesystem with a given Filesystem name.

Parameter

file_path: str - the UNIX directory that contains the files and directories to be listed. file_system_name: str - the name for the file system to be listed

rtype

json - A JSON containing the result of the operation

migrate_data_set(dataset_name: str, wait=False)

Migrates the data set.

Parameters

- **dataset_name** (str) – Name of the data set
- **wait** (bool) – If true, the function waits for completion of the request, otherwise the request is queued.

Return type

json - A JSON containing the result of the operation

mount_file_system(file_system_name, mount_point, options={}, encoding='utf-8')

Mounts a z/OS UNIX file system on a specified directory. Parameter ——— file_system_name: str - the name for the file system mount_point: str - mount point to be used for mounting the UNIX file system options: dict - A JSON of request body options

Return type

json - A JSON containing the result of the operation

recall_migrated_dataset(dataset_name: str, wait=False)

Recalls a migrated data set.

Parameters

- **dataset_name** (str) – Name of the data set
- **wait** (bool) – If true, the function waits for completion of the request, otherwise the request is queued

Return type

json - A JSON containing the result of the operation

rename_dataset(before_dataset_name: str, after_dataset_name: str)

Renames the data set.

Parameters

- **before_dataset_name** (str) – The source data set name.

- **after_dataset_name** (*str*) – New name for the source data set.

Return type

json - A JSON containing the result of the operation

rename_dataset_member(*dataset_name: str, before_member_name: str, after_member_name: str, enq=""*)

Renames the data set member.

Parameters

- **dataset_name** (*str*) – Name of the data set.
- **before_member_name** (*str*) – The source member name.
- **after_member_name** (*str*) – New name for the source member.
- **enq** (*str*) – Values can be SHRW or EXCLU. SHRW is the default for PDS members, EXCLU otherwise.

Return type

json - A JSON containing the result of the operation

unmount_file_system(*file_system_name, options={}, encoding='utf-8'*)

Unmounts a z/OS UNIX file system on a specified directory.

Parameter

file_system_name: str - the name for the file system options: dict - A JSON of request body options

rtype

json - A JSON containing the result of the operation

upload_file_to_dsn(*input_file, dataset_name, encoding='utf-8'*)

Upload contents of a given file and uploads it to a dataset.

upload_file_to_uss(*input_file, filepath_name, encoding='utf-8'*)

Upload contents of a given file and uploads it to UNIX file

write_to_dsn(*dataset_name, data, encoding='utf-8'*)

Write content to an existing dataset.

Returns

A JSON containing the result of the operation

Return type

json

write_to_uss(*filepath_name, data, encoding='utf-8'*)

Write content to an existing UNIX file. :returns: A JSON containing the result of the operation :rtype: json

Exceptions classes

InvalidPermsOption

class `zos_files.zowe.zos_files_for_zowe_sdk.exceptions.InvalidPermsOption(value: int)`
 Class used to represent an invalid permission value.

MaxAllocationQuantityExceeded

class `zos_files.zowe.zos_files_for_zowe_sdk.exceptions.MaxAllocationQuantityExceeded`
 Class used to represent an invalid allocation quantity.

3.4 z/OS Jobs

Contains the z/OSMF Jobs REST API functionalities.

3.4.1 Installing

```
pip install zowe.zos_jobs_for_zowe_sdk
```

3.4.2 Reference

`zowe.zos_jobs_for_zowe_sdk`

Jobs

class `zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs(connection)`
 Class used to represent the base z/OSMF Jobs API.

connection

Connection object

cancel_job(*jobname: str, jobid: str, modify_version='2.0'*)

Cancels the a job

Parameters

- **jobname** (*str*) – The name of the job
- **jobid** (*str*) – The job id on JES
- **modify_version** (*str*) – Default (“2.0”) specifies that the request is to be processed synchronously. For asynchronous processing - change the value to “1.0”

Returns

A JSON object containing the result of the request execution

Return type

`response_json`

change_job_class(*jobname: str, jobid: str, class_name: str, modify_version='2.0'*)

Changes the job class

Parameters

- **jobname** (*str*) – The name of the job
- **jobid** (*str*) – The job id on JES
- **modify_version** (*str*) – Default (“2.0”) specifies that the request is to be processed synchronously. For asynchronous processing - change the value to “1.0”

Returns

A JSON object containing the result of the request execution

Return type

response_json

delete_job(*jobname, jobid, modify_version='2.0'*)

Delete the given job on JES.

Parameters

- **jobname** (*str*) – The name of the job
- **jobid** (*str*) – The job id on JES
- **modify_version** (*str*) – Default (“2.0”) specifies that the request is to be processed synchronously. For asynchronous processing - change the value to “1.0”

Returns

A JSON object containing the result of the request execution

Return type

response_json

get_jcl_text(*correlator*)

Retrieve the input JCL text for job with specified correlator :param correlator: The correlator of the job. This is the value of the key ‘job-correlator’ in the status json :type correlator: str

Returns

A JSON object containing the result of the request execution

Return type

json

get_job_output_as_files(*status, output_dir*)

This method will get all the spool files as well as the submitted jcl text in separate files in the specified output directory. The structure will be as follows:

–<output directory> | file: jcl.txt | dir: jobname

dir: jobid | dir: stepname | file: spool file <nn> ...

Parameters

- **status** (*json*) – The response json describing the job to be used. (i.e. from the last get_status call)
- **output_dir** (*str*) – The output directory where the output files will be stored. The directory does not have to exist yet

Returns

A JSON object containing the result of the request execution

Return type

json

get_job_status(*jobname*, *jobid*)

Retrieve the status of a given job on JES.

Parameters

- **jobname** (*str*) – The name of the job
- **jobid** (*str*) – The job id on JES

Returns

A JSON object containing the status of the job on JES

Return type

response_json

get_spool_file_contents(*correlator*, *id*)

Retrieve the contents of a single spool file from a job

Parameters

- **correlator** (*str*) – The correlator of the job. This is the value of the key ‘job-correlator’ in the status json
- **id** (*str*) – The id number of the spool file. This is returned in the get_spool_files return json

Returns

A JSON object containing the result of the request execution

Return type

json

get_spool_files(*correlator*)

Retrieve the spool files for a job identified by the correlator.

Parameters

correlator (*str*) – The correlator of the job. This is the value of the key ‘job-correlator’ in the status json

Returns

A JSON object containing the result of the request execution

Return type

json

hold_job(*jobname: str*, *jobid: str*, *modify_version='2.0'*)

Hold the given job on JES

Parameters

- **jobname** (*str*) – The name of the job
- **jobid** (*str*) – The job id on JES
- **modify_version** (*str*) – Default (“2.0”) specifies that the request is to be processed synchronously. For asynchronous processing - change the value to “1.0”

Returns

A JSON object containing the result of the request execution

Return type

response_json

list_jobs(owner=None, prefix='*', max_jobs=1000, user_correlator=None)

Retrieve list of jobs on JES based on the provided arguments.

Parameters

- **owner** (*str*, *optional*) – The job owner (default is zosmf user)
- **prefix** (*str*, *optional*) – The job name prefix (default is *)
- **max_jobs** (*int*, *optional*) – The maximum number of jobs in the output (default is 1000)
- **user_correlator** (*str*, *optional*) – The z/OSMF user correlator attribute (default is None)

Returns

A JSON object containing a list of jobs on JES queue based on the given parameters

Return type

json

release_job(jobname: str, jobid: str, modify_version='2.0')

Release the given job on JES

Parameters

- **jobname** (*str*) – The name of the job
- **jobid** (*str*) – The job id on JES
- **modify_version** (*str*) – Default (“2.0”) specifies that the request is to be processed synchronously. For asynchronous processing - change the value to “1.0”

Returns

A JSON object containing the result of the request execution

Return type

response_json

submit_from_local_file(jcl_path)

Submit a job from local file.

This function will internally call the *submit_plaintext* function in order to submit the contents of the given input file

Parameters**jcl_path** (*str*) – Path to the local file where the JCL is located**Raises****FileNotFoundError** – If the local file provided is not found**Returns**

A JSON object containing the result of the request execution

Return type

json

submit_from_mainframe(jcl_path)

Submit a job from a given dataset.

Parameters**jcl_path** (*str*) – The dataset where the JCL is located

Returns

A JSON object containing the result of the request execution

Return type

json

submit_plaintext(*jcl*)

Submit a job from plain text input.

Parameters

jcl (*str*) – The plain text JCL to be submitted

Returns

A JSON object containing the result of the request execution

Return type

json

3.5 z/OS TSO

Contains the z/OSMF TSO REST API functionalities.

3.5.1 Installing

```
pip install zowe.zos_tso_for_zowe_sdk
```

3.5.2 Reference

zowe.zos_tso_for_zowe_sdk

Tso

class `zos_tso.zowe.zos_tso_for_zowe_sdk.Tso`(*connection*, *tso_profile=None*)

Class used to represent the base z/OSMF TSO API.

...

connection

Connection object

session_not_found

Constant for the session not found tso message id

end_tso_session(*session_key*)

Terminates an existing TSO session.

Parameters

session_key (*str*) – The session key of an existing TSO session

Returns

A string informing if the session was terminated successfully or not

Return type

str

issue_command(*command*)

Issues a TSO command.

This function will first initiate a TSO session, retrieve the session key, send the command and finally terminate the session

Parameters

command (*str*) – TSO command to be executed

Returns

A list containing the output from the TSO command

Return type

list

parse_message_ids(*response_json*)

Parse TSO response and retrieve only the message ids.

Parameters

response_json (*dict*) – The JSON containing the TSO response

Returns

A list containing the TSO response message ids

Return type

list

ping_tso_session(*session_key*)

Ping an existing TSO session and returns if it is still available.

Parameters

session_key (*str*) – The session key of an existing TSO session

Returns

A string informing if the ping was successful or not. Where the options are: 'Ping successful' or 'Ping failed'

Return type

str

retrieve_tso_messages(*response_json*)

Parse the TSO response and retrieve all messages.

Parameters

response_json (*dict*) – The JSON containing the TSO response

Returns

A list containing the TSO response messages

Return type

list

send_tso_message(*session_key, message*)

Send a command to an existing TSO session.

Parameters

- **session_key** (*str*) – The session key of an existing TSO session
- **message** (*str*) – The message/command to be sent to the TSO session

Returns

A non-normalized list from TSO containing the result from the command

Return type

list

start_tso_session(*proc=None, chset=None, cpage=None, rows=None, cols=None, rsize=None, acct=None*)

Start a TSO session.

Parameters

- **proc** (*str, optional*) – Proc parameter for the TSO session (default is “IZUFPROC”)
- **chset** (*str, optional*) – Chset parameter for the TSO session (default is “697”)
- **cpage** (*str, optional*) – Cpage parameter for the TSO session (default is “1047”)
- **rows** (*str, optional*) – Rows parameter for the TSO session (default is “204”)
- **cols** (*str, optional*) – Cols parameter for the TSO session (default is “160”)
- **rsize** (*str, optional*) – Rsize parameter for the TSO session (default is “4096”)
- **str** (*acctL*) – Acct parameter for the TSO session (default is “DEFAULT”)
- **optional** – Acct parameter for the TSO session (default is “DEFAULT”)

Returns

The ‘servletKey’ key for the created session (if successful)

Return type

str

3.6 z/OSMF

Contains the z/OSMF REST API functionalities.

3.6.1 Installing

```
pip install zowe.zosmf_for_zowe_sdk
```

3.6.2 Reference

zowe.zosmf_for_zowe_sdk

Zosmf

class zosmf.zowe.zosmf_for_zowe_sdk.**Zosmf**(*connection*)

Class used to represent the base z/OSMF API.

connection

Connection object

get_info()

Return a JSON response from the GET request to z/OSMF info endpoint.

Returns

A JSON containing the z/OSMF Info REST API data

Return type

json

list_systems()

Return a JSON response from the GET request to z/OSMF info endpoint. :returns: Return a list of the systems that are defined to a z/OSMF instance :rtype: json

CLASSES

4.1 zowe.secrets_for_zowe_sdk

CONTRIBUTING

This document contains the contribution guidelines for the Zowe Client Python SDK.

Notice that the Python SDK is still in early development stages, meaning that major architectural changes might be made by the development team at any given moment. **For this reason, only bug fixes and documentation changes are being accepted as contribution at this moment** . This document will be updated as soon as the stable release of the SDK is published (v1.0.0)

- *Git branching model*
- *Pull requests*
- *Running Tests*
- *Building Docs*
- *Code standards*

5.1 Git branching model

This project follows the [Git flow](#) branching model.

5.2 Pull requests

Consider the following when you interact with pull requests:

- Pull request reviewers should be assigned to a same-team member.
- Pull requests should remain open for at least 24 hours, or until close of the business next business day (accounting for weekends and holidays).
- Anyone can comment on a pull request to request delay on merging or to get questions answered.

5.3 Running Tests

The project's test suite can be run with the python test runner, *pytest*.

All test and regular dependencies are included here:

```
pip install -r requirements.txt
```

Commands for running all unit/integration tests from their respective folder:

```
pytest tests/unit
```

```
pytest tests/integration
```

More information on pytest's usage can be found [here](#).

In order to run integration tests, you will need to have a mainframe account and team profile configuration files properly set up.

Information on creating team profile configuration files can be found [here](#).

You will also need to update the [zowe.config.json](#) file with the necessary information.

5.4 Building Docs

The project's documentation is published on [ReadTheDocs.io](#).

To build the docs from source locally, you need these prerequisites:

- [Sphinx](#) - Python Documentation Generator
- Python packages - *pip install -r docs/requirements.txt*

Run *make html* in the docs directory to generate HTML files in “docs/build/html” that can be previewed in your browser.

Docs are generated from reStructuredText (.rst) files in “docs/source” and Python docstrings in the source code which also use reST markup. A quick reference for reStructuredText markup can be found [here](#).

5.5 Code standards

This project follows the [PEP 8](#) style guide.

INDEX

A

`ApiConnection` (class in `core.zowe.core_for_zowe_sdk`), 7
`autodiscover_config_dir()` (core.zowe.core_for_zowe_sdk.config_file.ConfigFile method), 12
`create_data_set()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 18
`create_default_data_set()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 18
`create_uss()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 18
`create_zFS_file_system()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 18

C

`cancel_job()` (zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs method), 23
`change_job_class()` (zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs method), 23
`config_appname` (core.zowe.core_for_zowe_sdk.ProfileManager property), 8
`config_dir` (core.zowe.core_for_zowe_sdk.ProfileManager property), 8
`config_filename` (core.zowe.core_for_zowe_sdk.ProfileManager property), 8
`config_filepath` (core.zowe.core_for_zowe_sdk.ProfileManager property), 8
`ConfigFile` (class in `core.zowe.core_for_zowe_sdk.config_file`), 11
`ConfigNotFoundWarning` (class in `core.zowe.core_for_zowe_sdk.custom_warnings`), 13
`connection` (zos_console.zowe.zos_console_for_zowe_sdk.Console attribute), 16
`connection` (zos_files.zowe.zos_files_for_zowe_sdk.Files attribute), 17
`connection` (zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs attribute), 23
`connection` (zos_tso.zowe.zos_tso_for_zowe_sdk.Tso attribute), 27
`connection` (zosmf.zowe.zosmf_for_zowe_sdk.Zosmf attribute), 29
`Console` (class in `zos_console.zowe.zos_console_for_zowe_sdk`), 16
`copy_dataset_or_member()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 17
`copy_uss_to_dataset()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 17

D

`data` (core.zowe.core_for_zowe_sdk.config_file.Profile attribute), 13
`delete_data_set()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 18
`delete_job()` (zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs method), 24
`delete_migrated_data_set()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 18
`delete_uss()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 18
`delete_zFS_file_system()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 19
`download_binary_dsn()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 19
`download_dsn()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 19
`download_uss()` (zos_files.zowe.zos_files_for_zowe_sdk.Files method), 19
`end_tso_session()` (zos_tso.zowe.zos_tso_for_zowe_sdk.Tso method), 27

E

`end_tso_session()` (zos_tso.zowe.zos_tso_for_zowe_sdk.Tso method), 27

F

`FileNotFound` (class in `core.zowe.core_for_zowe_sdk.exceptions`), 14

`Files` (class in `zos_files.zowe.zos_files_for_zowe_sdk`), 17

`find_profile()` (`core.zowe.core_for_zowe_sdk.config_file.ConfigFile` method), 12

G

`get_dsn_binary_content()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 19

`get_dsn_binary_content_streamed()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 19

`get_dsn_content()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 19

`get_dsn_content_streamed()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 20

`get_env()` (`core.zowe.core_for_zowe_sdk.ProfileManager` static method), 8

`get_file_content()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 20

`get_file_content_streamed()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 20

`get_highest_priority_layer()` (`core.zowe.core_for_zowe_sdk.ProfileManager` method), 8

`get_info()` (`zozmf.zowe.zozmf_for_zowe_sdk.Zozmf` method), 29

`get_jcl_text()` (`zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs` method), 24

`get_job_output_as_files()` (`zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs` method), 24

`get_job_status()` (`zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs` method), 25

`get_profile()` (`core.zowe.core_for_zowe_sdk.config_file.ConfigFile` method), 12

`get_profile()` (`core.zowe.core_for_zowe_sdk.ProfileManager` static method), 8

`get_profile_name_from_path()` (`core.zowe.core_for_zowe_sdk.config_file.ConfigFile` method), 12

`get_profile_path_from_name()` (`core.zowe.core_for_zowe_sdk.config_file.ConfigFile` method), 12

`get_profilename_from_profiletype()` (`core.zowe.core_for_zowe_sdk.config_file.ConfigFile` method), 12

`get_response()` (`zos_console.zowe.zos_console_for_zowe_sdk.Console` method), 16

`get_spool_file_contents()` (`zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs` method), 25

`get_spool_files()` (`zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs` method), 25

H

`hold_job()` (`zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs` method), 25

`host_url` (`core.zowe.core_for_zowe_sdk.ApiConnection` attribute), 7

I

`init_from_file()` (`core.zowe.core_for_zowe_sdk.config_file.ConfigFile` method), 12

`InvalidPermsOption` (class in `zos_files.zowe.zos_files_for_zowe_sdk.exceptions`), 23

`InvalidRequestMethod` (class in `core.zowe.core_for_zowe_sdk.exceptions`), 14

`ISession` (class in `core.zowe.core_for_zowe_sdk.session`), 15

`issue_command()` (`zos_console.zowe.zos_console_for_zowe_sdk.Console` method), 16

`issue_command()` (`zos_tso.zowe.zos_tso_for_zowe_sdk.Tso` method), 27

J

`Jobs` (class in `zos_jobs.zowe.zos_jobs_for_zowe_sdk`), 23

L

`list_dsn()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 20

`list_dsn_members()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 20

`list_files()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 20

`list_jobs()` (`zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs` method), 26

`list_systems()` (`zozmf.zowe.zozmf_for_zowe_sdk.Zozmf` method), 30

`list_unix_file_systems()` (`zos_files.zowe.zos_files_for_zowe_sdk.Files` method), 21

`load()` (`core.zowe.core_for_zowe_sdk.ProfileManager` method), 9

`load()` (`core.zowe.core_for_zowe_sdk.ZozmfProfile` method), 11

`load_profile_properties()` (`core.zowe.core_for_zowe_sdk.config_file.ConfigFile` method), 12

load_secure_props()
(core.zowe.core_for_zowe_sdk.CredentialManager
static method), 8

M

MaxAllocationQuantityExceeded (class in
zos_files.zowe.zos_files_for_zowe_sdk.exceptions), 23

migrate_data_set() (zos_files.zowe.zos_files_for_zowe_sdk.Files
method), 21

missing_secure_props
(core.zowe.core_for_zowe_sdk.config_file.Profile
attribute), 13

MissingConnectionArgs (class in
core.zowe.core_for_zowe_sdk.exceptions),
14

mount_file_system()
(zos_files.zowe.zos_files_for_zowe_sdk.Files
method), 21

N

name (core.zowe.core_for_zowe_sdk.config_file.Profile
attribute), 13

P

parse_message_ids()
(zos_tso.zowe.zos_tso_for_zowe_sdk.Tso
method), 28

password (core.zowe.core_for_zowe_sdk.ApiConnection
attribute), 7

perform_request() (core.zowe.core_for_zowe_sdk.RequestHandler
method), 10

perform_streamed_request()
(core.zowe.core_for_zowe_sdk.RequestHandler
method), 10

ping_tso_session() (zos_tso.zowe.zos_tso_for_zowe_sdk.Tso
method), 28

Profile (class in core.zowe.core_for_zowe_sdk.config_file), 13

profile_name (core.zowe.core_for_zowe_sdk.ZosmfProfile
attribute), 11

ProfileManager (class in
core.zowe.core_for_zowe_sdk), 8

ProfileNotFound (class in
core.zowe.core_for_zowe_sdk.exceptions),
14

ProfileNotFoundWarning (class in
core.zowe.core_for_zowe_sdk.custom_warnings),
13

ProfileParsingWarning (class in
core.zowe.core_for_zowe_sdk.custom_warnings),
14

profiles_dir (core.zowe.core_for_zowe_sdk.ZosmfProfileset_property), 11

R

recall_migrated_dataset()
(zos_files.zowe.zos_files_for_zowe_sdk.Files
method), 21

release_job() (zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs
method), 26

rename_dataset() (zos_files.zowe.zos_files_for_zowe_sdk.Files
method), 21

rename_dataset_member()
(zos_files.zowe.zos_files_for_zowe_sdk.Files
method), 22

RequestFailed (class in
core.zowe.core_for_zowe_sdk.exceptions),
14

RequestHandler (class in
core.zowe.core_for_zowe_sdk), 10

retrieve_tso_messages()
(zos_tso.zowe.zos_tso_for_zowe_sdk.Tso
method), 28

S

save() (core.zowe.core_for_zowe_sdk.config_file.ConfigFile
method), 12

save() (core.zowe.core_for_zowe_sdk.ProfileManager
method), 9

save_secure_props()
(core.zowe.core_for_zowe_sdk.CredentialManager
static method), 8

schema_list() (core.zowe.core_for_zowe_sdk.config_file.ConfigFile
method), 12

SecureApi (class in core.zowe.core_for_zowe_sdk), 11

SecureProfileLoadFailed (class in
core.zowe.core_for_zowe_sdk.exceptions),
15

SecurePropsNotFoundWarning (class in
core.zowe.core_for_zowe_sdk.custom_warnings),
14

SecureValuesNotFound (class in
core.zowe.core_for_zowe_sdk.exceptions),
15

send_tso_message() (zos_tso.zowe.zos_tso_for_zowe_sdk.Tso
method), 28

Session (class in core.zowe.core_for_zowe_sdk.session),
15

session_arguments (core.zowe.core_for_zowe_sdk.RequestHandler
attribute), 10

session_not_found (zos_tso.zowe.zos_tso_for_zowe_sdk.Tso
attribute), 27

set_profile() (core.zowe.core_for_zowe_sdk.config_file.ConfigFile
method), 13

set_profile() (core.zowe.core_for_zowe_sdk.ProfileManager
method), 9

set_property() (core.zowe.core_for_zowe_sdk.config_file.ConfigFile
method), 13

[set_property\(\)](#) (*core.zowe.core_for_zowe_sdk.ProfileManager* *method*), [9](#)
[ZosmfProfile](#) (*class in core.zowe.core_for_zowe_sdk*), [11](#)
[ssl_verification\(\)](#) (*core.zowe.core_for_zowe_sdk.ApiConnection* *attribute*), [7](#)
[start_tso_session\(\)](#) (*zos_tso.zowe.zos_tso_for_zowe_sdk.Tso* *method*), [29](#)
[submit_from_local_file\(\)](#) (*zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs* *method*), [26](#)
[submit_from_mainframe\(\)](#) (*zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs* *method*), [26](#)
[submit_plaintext\(\)](#) (*zos_jobs.zowe.zos_jobs_for_zowe_sdk.Jobs* *method*), [27](#)

T

[Tso](#) (*class in zos_tso.zowe.zos_tso_for_zowe_sdk*), [27](#)

U

[UnexpectedStatus](#) (*class in core.zowe.core_for_zowe_sdk.exceptions*), [15](#)
[unmount_file_system\(\)](#) (*zos_files.zowe.zos_files_for_zowe_sdk.Files* *method*), [22](#)
[UnsupportedAuthType](#) (*class in core.zowe.core_for_zowe_sdk.exceptions*), [15](#)
[upload_file_to_dsn\(\)](#) (*zos_files.zowe.zos_files_for_zowe_sdk.Files* *method*), [22](#)
[upload_file_to_uss\(\)](#) (*zos_files.zowe.zos_files_for_zowe_sdk.Files* *method*), [22](#)
[user](#) (*core.zowe.core_for_zowe_sdk.ApiConnection* *attribute*), [7](#)
[user_config_dir](#) (*core.zowe.core_for_zowe_sdk.ProfileManager* *property*), [9](#)

V

[valid_methods](#) (*core.zowe.core_for_zowe_sdk.RequestHandler* *attribute*), [10](#)
[validate_schema\(\)](#) (*core.zowe.core_for_zowe_sdk.config_file.ConfigFile* *method*), [13](#)

W

[write_to_dsn\(\)](#) (*zos_files.zowe.zos_files_for_zowe_sdk.Files* *method*), [22](#)
[write_to_uss\(\)](#) (*zos_files.zowe.zos_files_for_zowe_sdk.Files* *method*), [22](#)

Z

[Zosmf](#) (*class in zosmf.zowe.zosmf_for_zowe_sdk*), [29](#)