Zowe Client Python SDK

Release 1.0.0-dev26

Contributors to the Zowe Project

Jun 19, 2026

CONTENTS:

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.

CHAPTER

ONE

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.

CHAPTER

TWO

GETTING STARTED

Review the requirements and installation instructions to get started using the Zowe Client Python SDK.

2.1 Requirements

Zowe Client Python SDK requires the following:

- **Python** 3.10+ to run your script
- (optional) **Sphinx** to build project documentation
- (optional) **Enchant** to build project documentation

2.2 Installation

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

- Install **all Zowe packages** to install everything under the *zowe* namespace in PyPi.
- Install a **single subpackage** for a smaller installation.

To install all Zowe SDK packages using *pip*:

```
pip install -U --pre zowe-python-sdk-bundle
```

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.

2.3 First steps

After you install the package in your project, integrate the SDK in your script:

1. Import the class for the required sub-package in order to call the individual SDK method and run plug-in commands.

   For example, the *Console* class must be imported for z/OS Console commands to be issued.

2. Create a dictionary to add connection information to communicate 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>",
    # "rejectUnauthorized": True, # Set to False to disable SSL verification
    # "basePath": "", # Define base path if using Zowe API ML (e.g. "/
↪ibmzosmf/api/v1" for z/OSMF)
    # "protocol": "https", # Include the protocol if different from the␣
↪default (https)
}

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)
```

2.4 Advanced steps

- **Use a custom Certificate Authority if working in a restricted environment.**
  The Python SDK supports the commonly used environmental variables *REQUESTS_CA_BUNDLE* and *CURL_CA_BUNDLE* to provide a certificate chain.

  You can also use the *SSL_CERT_FILE* environmental variable in project-level configurations.

CHAPTER

THREE

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: str*, *user: str*, *password: str*, *ssl_verification: bool = True*)

Class used to represent a connection with a REST API.

Parameters

- **host_url** (*str*) – The base url of the rest api host
- **user** (*str*) – The user of the rest api
- **password** (*str*) – The password for the user
- **ssl_verification** (*bool*) – Options for ssl verification. True by default.

Raises

MissingConnectionArgs – Missing connection argument.

CredentialManager

class core.zowe.core_for_zowe_sdk.**CredentialManager**

A class including static functions for managing credentials.

static load_secure_props() → None

Load secure_props stored for the given config file.

if keyring is not initialized, set empty value

Raises
SecureProfileLoadFailed – Fail to load secure profile

static save_secure_props() → None

Set secure_props for the given config file.

Log

class core.zowe.core_for_zowe_sdk.**Log**

Class used to represent a logger.

dirname

Path where the log file is saved

Type
str

file_handler

Shared FileHandler object for managing log file output

Type
logging.FileHandler

console_handler

Shared StreamHandler object for managing log console output

Type
logging.StreamHandler

file_output

Specifies whether log messages would be saved to a file. True by default.

Type
bool

console_output

Specifies whether log messages would be printed out on console. True by default.

Type
bool

loggers

The set of all loggers

Type
set[logging.Logger]

static close(*logger: Logger*) → None

Disable a logger.

Parameters
logger (*logging.Logger*) – The logger to be turned off

static close_all() → None

Disable all loggers.

static close_console_output() → None

Turn off log output to console.

static close_file_output() → None

Turn off log output to a file.

static open(*logger: Logger*) → None

Enable a logger.

Parameters
logger (*logging.Logger*) – The logger to be turned on

static open_all() → None

Enable all loggers.

static open_console_output() → None

Turn on log output to console.

static open_file_output() → None

Turn on log output to a file.

static register_logger(*name: str*) → Logger

Create and register a logger.

Parameters
name (*str*) – The name for the logger.

Returns
A Logger object named after the file where it is created.

Return type
logging.Logger

static set_all_logger_level(*level: int*) → None

Set display level for all loggers.

Parameters
level (*int*) – The intended logger level

static set_console_output_level(*level: int*) → None

Set the level for the console handler.

Parameters
level (*int*) – The intended console output level

static set_file_output_level(*level: int*) → None

Set the level for the file handler.

Parameters
level (*int*) – The intended file output level

ProfileManager

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

Class used to manage profiles.

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.

Parameters

- **appname** (*str*) – Name of the app

- **show_warnings** (*bool*) – Indicates whether warnings are shown

property config_appname: str

Return the application name.

Returns
The name of the application as configured in the current instance.

Return type
str

property config_dir: str | None

Return the folder path where the Zowe z/OSMF Team Project Config files are stored.

Returns
The directory path where the main project configuration files are located. This path can be None if not set.

Return type
Optional[str]

property config_filename: str

Return the filename for the Zowe z/OSMF Team Project Config.

Returns
The filename of the main project configuration file.

Return type
str

property config_filepath: str | None

Get the full filepath for the Zowe z/OSMF Team Project Config file.

Returns
Filepath of configuration file or None if the location or filename is not set.

Return type
Optional[str]

static get_env(*cfg:* ConfigFile, *cwd: str | None = None*) → dict[str, Any]

Map the env variables to the profile properties.

Parameters

- **cfg** (ConfigFile) – A config file that contains the schema properties.
- **cwd** (*Optional[str]*) – Path of current working directory.

Returns
Containing profile properties from env variables (prop: value).

Return type
dict[str, Any]

get_highest_priority_layer(*json_path: str*) → *ConfigFile* | None

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

Parameters
json_path (*str*) – The path of the json.

Raises
FileNotFoundError – File is not found in given path.

Returns
The highest priority layer (configuration file) that contains the specified profile, or None if the profile is not found in any layer.

Return type
Optional[*ConfigFile*]

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

Retrieve a profile from the configuration file, optionally validating the schema.

Parameters

- **cfg** (*ConfigFile*) – The configuration file object which contains the profiles.
- **profile_name** (*Optional[str]*) – The name of the profile to retrieve. If None, the method attempts to fetch the profile based only on the type.
- **profile_type** (*Optional[str]*) – The type of the profile to retrieve. If None, the method attempts to fetch the profile based only on the name.
- **validate_schema** (*Optional[bool]*) – Whether to validate the profile against the schema present in the configuration file.

Returns
A NamedTuple containing the profile data, name, and any secure properties not found.

Return type
Profile

Raises

- **ValidationError** – If the instance is invalid under the provided schema.
- **SchemaError** – If the provided schema itself is invalid.
- **UndefinedTypeCheck** – If a type checker is asked to check a type it does not have registered.
- **UnknownType** – If an unknown type is found in the schema.
- **FormatError** – If validating a format in the configuration fails.

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[str, Any]

Load connection details from a team config profile.

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.

Parameters

- **profile_name** (*Optional[str]*) – The name of the profile to load. If None, profiles are loaded based only on profile type.
- **profile_type** (*Optional[str]*) – The type of the profile to load, e.g., 'zosmf', 'zftp'. If None, profiles are loaded based only on name.
- **check_missing_props** (*bool*) – Flag to indicate whether to check for missing secure properties.

- **validate_schema** (*Optional[bool]*) – Whether to validate the loaded profile against the schema defined in the configuration.
- **override_with_env** (*Optional[bool]*) – If True, overrides profile properties with values from environment variables.

Raises

- ***ProfileNotFound*** – If both profile_name and profile_type are not provided, indicating which profile to load.
- ***SecureValuesNotFound*** – If any secure properties are required but not found or loaded.

Returns
A dictionary containing the merged connection details from all relevant profiles.

Return type
dict[str, Any]

save() → None

Save the layers (configuration files) to disk.

set_profile(*profile_path: str*, *profile_data: dict[str, Any]*) → 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[str, Any]*) – The data of the profile to set.

set_property(*json_path: str*, *value: str*, *secure: bool | None = 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** (*Optional[bool]*) – If True, the property will be stored securely. Default is None.

property user_config_dir: str | None

Return the folder path where the Zowe z/OSMF User Project Config files are stored.

Returns
The directory path where the user-specific project configuration files are located.

Return type
Optional[str]

RequestHandler

class core.zowe.core_for_zowe_sdk.**RequestHandler**(*session_arguments: dict[str, Any]*, *logger_name: str = 'core.zowe.core_for_zowe_sdk.request_handler'*)

Class used to handle HTTP/HTTPS requests.

Parameters

- **session_arguments** (*dict[str, Any]*) – Zowe SDK session arguments
- **logger_name** (*str*) – The logger name of the modules calling request handler

__del__() → None

Clean up the REST session object once it is no longer needed anymore.

perform_request(*method: str*, *request_arguments: dict[str, Any]*, *expected_code: list[int] = [200]*, *stream: bool = False*) → str | bytes | Response | dict[str, Any] | None

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[str, Any]*) – The dictionary containing the required arguments for the execution of the request
- **expected_code** (*list[int]*) – The list containing the acceptable response codes (default is [200])
- **stream** (*bool*) – The boolean value whether the request is stream

Returns
normalized request response in json (dictionary)

Return type
Union[str, bytes, Response, dict[str, Any], None]

SdkApi

class core.zowe.core_for_zowe_sdk.**SdkApi**(*profile: dict[str, Any]*, *default_url: str*, *logger_name: str = 'core.zowe.core_for_zowe_sdk.sdk_api'*, *log: bool = True*)

Abstract class used to represent the base SDK API.

Parameters

- **profile** (*dict[str, Any]*) – Profile information in json (dict) format
- **default_url** (*str*) – Default url used for session
- **logger_name** (*str*) – Name of the logger (same as the filename by default)
- **log** (*bool*) – Flag to disable logger

__enter__() → *SdkApi*

Return the SdkApi instance.

__exit__(*exc_type: Type[BaseException] | None*, *exception: BaseException | None*, *traceback: object | None*) → None

Delete the request handler before exit.

ZosmfProfile

class core.zowe.core_for_zowe_sdk.**ZosmfProfile**(*profile_name: str*)

Class used to represent a Zowe z/OSMF profile.

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.

Parameters
profile_name (*str*) – Zowe z/OSMF profile name

load() → *ApiConnection*

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

Returns
z/OSMF connection object

Return type
ApiConnection

property profiles_dir: str

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

Returns
the os path for the Zowe z/OSMF profiles

Return type
str

Config File classes

ConfigFile

class core.zowe.core_for_zowe_sdk.config_file.**ConfigFile**(*type: str*, *name: str*, *_location: str | None = None*, *profiles: dict[str, ~typing.Any] | None=None*, *defaults: dict[str, ~typing.Any] | None=None*, *schema_property: dict[str, ~typing.Any] | None=None*, *secure_props: dict[str, ~typing.Any] | None=None*, *jsonc: dict[str, ~typing.Any] | None=None*, *_missing_secure_props: list[str] = <factory>*, *_ConfigFile__suppress_config_file_warnings: bool | None = True*)

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.

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

Raises
FileNotFoundError – Cannot find file in directory.

find_profile(*path: str*, *profiles: dict[str, Any]*) → dict[str, Any] | None

Find a profile at a specified location from within a set of nested profiles.

Parameters

- **path** (*str*) – The location to look for the profile
- **profiles** (*dict[str, Any]*) – A dict of nested profiles

Returns
The profile object that was found, or None if not found

Return type
Optional[dict[str, Any]]

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.

Parameters

- **profile_name** (*Optional[str]*) – Name of the profile
- **profile_type** (*Optional[str]*) – Type of the profile
- **validate_schema** (*Optional[bool]*) – True if validation is preferred

Raises
ProfileNotFound – Cannot find profile

Returns
Returns a namedtuple called Profile

Return type
Profile

get_profile_name_from_path(*path: str*) → str

Get the name of the profile from the given path.

Parameters
path (*str*) – The location to look for the profile

Returns
Returns the profile name

Return type
str

get_profile_path_from_name(*short_path: str*) → str

Get the path of the profile from the given name.

Parameters
short_path (*str*) – Partial path of profile

Returns
Returns the full profile path

Return type
str

get_profilename_from_profiletype(*profile_type: str*) → str

Return profilename from given profiletype as defined in the team config profile.

Parameters
profile_type (*str*) – Type of the profile

Returns
The exact profilename of the profile to load from the mentioned type.

Return type
str

Raises
ProfileNotFound – Cannot find profile

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

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

Parameters
validate_schema (*Optional[bool]*) – True if validation is preferred, false otherwise

load_profile_properties(*profile_name: str*) → dict[str, Any]

Load profile properties given profile_name including secure properties.

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

Parameters
profile_name (*str*) – Name of the profile

Returns
Object containing profile properties

Return type
dict[str, Any]

Raises
ValueError – Profile cannot be None

save(*update_secure_props: bool | None = True*) → None

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

Parameters
update_secure_props (*Optional[bool]*) – If True, the secure properties will be stored in the vault. Default is True.

Raises
ValueError – Filepath must be set and valid.

schema_list(*cwd: str | None = None*) → list[dict[str, Any]]

Load the schema properties in a sorted order according to the priority.

Parameters
cwd (*Optional[str]*) – current working directory

Returns
properties from schema

Return type
list[dict[str, Any]]

set_profile(*profile_path: str*, *profile_data: dict[str, Any]*) → 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[str, Any]*) – The data to be set for the profile.

set_property(*json_path: str*, *value: str*, *secure: bool | None = 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** (*Optional[bool]*) – If True, the property will be stored securely. Default is None.

suppress_config_warnings(*value: bool*) → None

Suppress warnings in config files.

Parameters

value (*bool*) – Warnings are shown or not

validate_schema() → None

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

Profile

class core.zowe.core_for_zowe_sdk.config_file.**Profile**(*data: dict[str, Any] = {}*, *name: str = ''*, *missing_secure_props: list[str] = []*)

Class to represent a profile.

data: dict[str, Any]

Alias for field number 0

missing_secure_props: list[str]

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: str*)

A warning that is raised when a configuration file is not found.

Parameters

message (*str*) – A human-readable string describing the warning.

__str__() → str

Return a string representation of the warning message.

Returns

a string representation of the warning message

Return type

str

ProfileNotFoundWarning

class core.zowe.core_for_zowe_sdk.custom_warnings.**ProfileNotFoundWarning**(*message: str*)

A warning that is raised when a user profile cannot be found.

Parameters
message (*str*) – A string describing the warning.

__str__() → str

Return a string representation of the warning message.

Returns
a string representation of the warning message

Return type
str

ProfileParsingWarning

class core.zowe.core_for_zowe_sdk.custom_warnings.**ProfileParsingWarning**(*message: str*)

A warning that is raised when there is an error while parsing a user profile.

Parameters
message (*str*) – A human-readable string describing the warning.

__str__() → str

Return a string representation of the warning message.

Returns
a string representation of the warning message

Return type
str

SecurePropsNotFoundWarning

class core.zowe.core_for_zowe_sdk.custom_warnings.**SecurePropsNotFoundWarning**(*message: str*)

A warning that is raised when secure properties are not found.

Parameters
message (*str*) – A human-readable string describing the warning.

__str__() → str

Return a string representation of the warning message.

Returns
a string representation of the warning message

Return type
str

Exceptions classes

FileNotFound

class core.zowe.core_for_zowe_sdk.exceptions.**FileNotFound**(*input_path: str*)

Class used to represent a file not found exception.

Parameters
input_path (*str*) – The invalid input path

InvalidRequestMethod

class core.zowe.core_for_zowe_sdk.exceptions.**InvalidRequestMethod**(*input_method: str*)

Class used to represent an invalid request method exception.

Parameters

input_method (*str*) – The invalid HTTP method used

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.

Parameters

- **profile_name** (*str*) – The name of the profile it failed to load
- **error_msg** (*str*) – The error message received while trying to load the profile

RequestFailed

class core.zowe.core_for_zowe_sdk.exceptions.**RequestFailed**(*status_code: int*, *request_output: str*)

Class used to represent a request failure exception.

Parameters

- **status_code** (*int*) – The status code from the failed request
- **request_output** (*str*) – The output from the request

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.

Parameters

- **profile_name** (*str*) – The name of the profile it failed to load
- **error_msg** (*str*) – The error message received while trying to load the profile

SecureValuesNotFound

class core.zowe.core_for_zowe_sdk.exceptions.**SecureValuesNotFound**(*values: set[str]*)

Class used to represent a profile load failure exception.

Parameters

values (*set[str]*) – The list of secure values not found

UnexpectedStatus

class core.zowe.core_for_zowe_sdk.exceptions.**UnexpectedStatus**(*expected: list[int]*, *received: int*, *request_output: str*)

Class used to represent an unexpected request response status exception.

Parameters

- **expected** (*list[int]*) – The list of expected status code
- **received** (*int*) – The received status code
- **request_output** (*str*) – The output from the request

UnsupportedAuthType

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

Class used to represent an unsupported authentication type exception.

Parameters

auth_type (*str*) – The type of authentication on the session

Session classes

ISession

class core.zowe.core_for_zowe_sdk.session.**ISession**(*host: str*, *port: int = 443*, *reject_unauthorized: bool = True*, *user: str | None = None*, *password: str | None = None*, *protocol: str = 'https'*, *base_path: str | None = None*, *type: str | None = None*, *token_type: str | None = None*, *token_value: str | None = None*, *cert: tuple[str, str] | None = None*)

Class to represent session parameters.

Session

class core.zowe.core_for_zowe_sdk.session.**Session**(*props: dict[str, Any]*)

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

Parameters

props (*dict[str, Any]*) – Profile and properties

Raises

ValueError – Exception thrown when cert key is not provided

property **host_url: str**

Return the formatted host URL.

Returns

the formatted host URL

Return type

str

load() → *ISession*

Load a ISession object.

Returns
A custom ISession object.

Return type
ISession

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: dict[str, Any]*, *log: bool = True*)

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

Parameters

- **connection** (*dict[str, Any]*) – A profile in dict (json) format
- **log** (*bool*) – Flag to disable logger

get_response(*response_key: str*, *console: str | None = None*) → ConsoleResponse

Collect outstanding synchronous z/OS Console response messages.

Parameters

- **response_key** (*str*) – The command response key from the Issue Command request.
- **console** (*Optional[str]*) – The console that should be used to get the command response.

Returns
A JSON containing the response to the command

Return type
ConsoleResponse

issue_command(*command: str*, *console: str | None = None*) → IssueCommandResponse

Issues a command on z/OS Console.

Parameters

- **command** (*str*) – The z/OS command to be executed
- **console** (*Optional[str]*) – Name of 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
IssueCommandResponse

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

BaseFilesApi

class zos_files.zowe.zos_files_for_zowe_sdk.**BaseFilesApi**(*profile: dict[str, Any]*, *log: bool = True*)

Extends the SdkApi class to support headers specific to z/OSMF Files APIs.

Parameters

- **profile** (*dict[str, Any]*) – Profile information in json (dict) format
- **log** (*bool*) – Flag to disable logger

FileSystems

class zos_files.zowe.zos_files_for_zowe_sdk.**FileSystems**(*connection: dict[str, Any]*, *log: bool = True*)

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

It includes all operations related to file systems.

Parameters

- **connection** (*dict[str, Any]*) – A profile for connection in dict (json) format
- **log** (*bool*) – Flag to disable logger

create(*file_system_name: str*, *options: dict[str, Any] = {}*) → None

Create a z/OS UNIX zFS Filesystem.

Parameters

- **file_system_name** (*str*) – Name of the file system
- **options** (*dict[str, Any]*) – Specifies file system attributes

Raises

- ***MaxAllocationQuantityExceeded*** – Thrown when file system exceeds max allocation quantity
- ***InvalidPermsOption*** – Thrown when invalid permission option is provided

delete(*file_system_name: str*) → None

Delete a zFS Filesystem.

Parameters

file_system_name (*str*) – Name of the file system

list(*file_path_name: str | None = None*, *file_system_name: str | None = None*) → FileSystemListResponse

List all mounted filesystems.

It could also list the specific filesystem mounted at a given path, or the filesystem with a given Filesystem name.

Parameters

- **file_path_name** (*Optional[str]*) – USS directory that contains the files and directories to be listed
- **file_system_name** (*Optional[str]*) – Name of the file system to be listed

Returns
A JSON containing the result of the operation

Return type
FileSystemListResponse

mount(*file_system_name: str*, *mount_point: str*, *options: dict[str, Any] = {}*, *encoding: str = 'utf-8'*) → None

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

Parameters

- **file_system_name** (*str*) – Name for the file system
- **mount_point** (*str*) – Mount point to be used for mounting the UNIX file system
- **options** (*dict[str, Any]*) – A JSON of request body options
- **encoding** (*str*) – Specifies optional encoding name (e.g. IBM-1047)

unmount(*file_system_name: str*, *options: dict[str, Any] = {}*, *encoding: str = 'utf-8'*) → None

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

Parameters

- **file_system_name** (*str*) – Name for the file system
- **options** (*dict[str, Any]*) – A JSON of request body options
- **encoding** (*str*) – Specifies optional encoding name (e.g. IBM-1047)

Files

class zos_files.zowe.zos_files_for_zowe_sdk.**Files**(*connection: dict[str, Any]*, *log: bool = True*)

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

ds

A Datasets class object

Type
Datasets

uss

An USSFiles class object

Type
USSFiles

fs

A FileSystems class object

Type
FileSystems

Parameters

- **connection** (*dict[str, Any]*) – The z/OSMF connection object (generated by the ZoweSDK object)
- **log** (*bool*) – Flag to disable logger

copy_data_set_or_member(*from_dataset_name: str*, *to_dataset_name: str*, *from_member_name: str | None = None*, *volser: str | None = None*, *alias: bool | None = None*, *to_member_name: str | None = None*, *enq: str | None = None*, *replace: bool = False*) → None

Use ds.copy_data_set_or_member() instead of this deprecated function.

copy_uss_to_data_set(*from_filename: str*, *to_dataset_name: str*, *to_member_name: str | None = None*, *type: FileType = FileType.TEXT*, *replace: bool = False*) → None

Use ds.copy_uss_to_data_set() instead of this deprecated function.

create_data_set(*dataset_name: str*, *options: DatasetOption | None = None*) → None

Use ds.create() instead of this deprecated function.

create_default_data_set(*dataset_name: str*, *default_type: str*) → None

Use ds.create_default() instead of this deprecated function.

create_uss(*file_path: str*, *file_type: str*, *mode: str | None = None*) → None

Use uss.create() instead of this deprecated function.

create_zfs_file_system(*file_system_name: str*, *options: dict[str, Any] | None = None*) → None

Use fs.create() instead of this deprecated function.

delete_data_set(*dataset_name: str*, *volume: str | None = None*, *member_name: str | None = None*) → None

Use ds.delete() instead of this deprecated function.

delete_migrated_data_set(*dataset_name: str*, *purge: bool = False*, *wait: bool = False*) → None

Use ds.delete_migrated() instead of this deprecated function.

delete_uss(*filepath_name: str*, *recursive: bool = False*) → None

Use uss.delete() instead of this deprecated function.

delete_zfs_file_system(*file_system_name: str*) → None

Use fs.delete() instead of this deprecated function.

download_binary_dsn(*dataset_name: str*, *output_file: str*, *with_prefixes: bool = False*) → None

Use *ds.perform_download(content_type=ContentType.BINARY)* instead of this deprecated function.

download_dsn(*dataset_name: str*, *output_file: str*) → None

Use *ds.perform_download()* instead of this deprecated function.

download_uss(*file_path: str*, *output_file: str*, *binary: bool = False*) → None

Use *uss.perform_download()* instead of this deprecated function.

get_dsn_binary_content(*dataset_name: str*, *with_prefixes: bool = False*) → bytes

Use *ds.retrieve_content(content_type=ContentType.BINARY)* instead of this deprecated function.

get_dsn_binary_content_streamed(*dataset_name: str*, *with_prefixes: bool = False*) → Response

Get binary content from the provided data set as a streamed Response.

This function is deprecated. Use *ds.retrieve_content(content_type=ContentType.BINARY, as_stream=True)* instead.

get_dsn_content(*dataset_name: str*) → str | None

Use ds.get_content() instead of this deprecated function.

get_dsn_content_streamed(*dataset_name: str*) → Response

Use *ds.retrieve_content(as_stream=True)* instead of this deprecated function.

get_file_content(*filepath_name: str*) → str | None

Use *uss.retrieve_content()* instead of this deprecated function.

get_file_content_streamed(*file_path: str*, *binary: bool = False*) → Response

Use *uss.retrieve_content(as_stream=True)* instead of this deprecated function.

list_dsn(*name_pattern: str*, *return_attributes: bool = False*) → DatasetListResponse

Use ds.list() instead of this deprecated function.

list_dsn_members(*dataset_name: str*, *member_pattern: str | None = None*, *member_start: str | None = None*, *limit: int = 1000*, *attributes: str = 'member'*) → MemberListResponse

Use ds.list_members() instead of this deprecated function.

list_files(*path: str*) → Any

Use uss.list() instead of this deprecated function.

list_unix_file_systems(*file_path_name: str | None = None*, *file_system_name: str | None = None*) → FileSystemListResponse

Use fs.list() instead of this deprecated function.

migrate_data_set(*dataset_name: str*, *wait: bool = False*) → None

Use ds.migrate() instead of this deprecated function.

mount_file_system(*file_system_name: str*, *mount_point: str*, *options: dict[str, Any] = {}*, *encoding: str = 'utf-8'*) → None

Use fs.mount() instead of this deprecated function.

recall_migrated_data_set(*dataset_name: str*, *wait: bool = False*) → None

Use ds.recall_migrated() instead of this deprecated function.

rename_data_set(*before_dataset_name: str*, *after_dataset_name: str*) → None

Use ds.rename() instead of this deprecated function.

rename_data_set_member(*dataset_name: str*, *before_member_name: str*, *after_member_name: str*, *enq: str = ''*) → None

Use ds.rename_member() instead of this deprecated function.

unmount_file_system(*file_system_name: str*, *options: dict[str, Any] | None = None*, *encoding: str = 'utf-8'*) → None

Use fs.unmount() instead of this deprecated function.

upload_file_to_dsn(*input_file: str*, *dataset_name: str*, *encoding: str = 'utf-8'*, *binary: bool = False*) → None

Use *ds.perform_upload()* instead of this deprecated function.

upload_file_to_uss(*input_file: str*, *filepath_name: str*, *encoding: str = 'utf-8'*) → None

Use *uss.perform_upload()* instead of this deprecated function.

write_to_dsn(*dataset_name: str*, *data: str | bytes*, *encoding: str = 'utf-8'*) → None

Use ds.write() instead of this deprecated function.

write_to_uss(*filepath_name: str*, *data: str*, *encoding: str = 'utf-8'*) → None

Use uss.write() instead of this deprecated function.

USSFiles

class zos_files.zowe.zos_files_for_zowe_sdk.**USSFiles**(*connection: dict[str, Any]*, *log: bool = True*)

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

It includes all operations related to USS files.

Parameters

- **connection** (*dict[str, Any]*) – The z/OSMF connection object (generated by the ZoweSDK object)
- **log** (*bool*) – Flag to disable logger

create(*file_path: str*, *type: str*, *mode: str | None = None*) → None

Add a file or directory.

Parameters

- **file_path** (*str*) – Path of the file to add
- **type** (*str*) – Specify either "file" or "dir"
- **mode** (*Optional[str]*) – Unix permission string (e.g. *rwxr-xr-x*)

delete(*filepath_name: str*, *recursive: bool = False*) → None

Delete a file or directory.

Parameters

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

download(*file_path: str*, *output_file: str*, *binary: bool = False*, *file_encoding: str = 'IBM-1047'*, *receive_encoding: str = 'UTF-8'*) → None

Use *perform_download* instead of this deprecated function.

get_content(*filepath_name: str*, *file_encoding: str = 'IBM-1047'*, *receive_encoding: str = 'ISO8859-1'*) → str | None

Use *retrieve_content()* instead of this deprecated function.

get_content_streamed(*file_path: str*, *binary: bool = False*, *file_encoding: str = 'IBM-1047'*, *receive_encoding: str = 'ISO8859-1'*) → Response

Use *retrieve_content(as_stream=True)* instead of this deprecated function.

get_file_tag(*filepath_name: str*) → USSFileTag

Retrieve the file tag if specified for the filename. Raises exception if it is impossible to identify the tag info.

Parameters

filepath_name (*str*) – Path of the file

Returns
Tag info of a given file.

Return type
USSFileTag

list(*path: str*) → USSListResponse

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

Parameters
path (*str*) – Path to retrieve the list

Returns
A JSON with a list of file names matching the given pattern

Return type
USSListResponse

perform_download(*remote_file_path: str*, *local_file_path: str*, *content_type:* ContentType = *ContentType.TEXT*, *remote_file_encoding: str = 'IBM-1047'*, *receive_in_encoding: str = 'UTF-8'*) → None

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

Parameters

- **remote_file_path** (*str*) – Path of the file to be downloaded
- **local_file_path** (*str*) – Name of the file to be saved locally
- **content_type** (ContentType, *optional*) – Specifies the content type to fetch ("text" or "binary", "text" by default)
- **remote_file_encoding** (*str, optional*) – Encoding file content originally in (to convert from; by default, it is always being converted from "IBM-1047")
- **receive_in_encoding** (*str, optional*) – Encoding to convert file content to (to convert to; by default, it is always being converted to "UTF-8" during download). Ignored when "binary" is True

Raises

- **TypeError** – Thrown when the *retrieve_content* request does not return a valid Response object.
- **ValueError** – Content type must be either ContenType.TEXT or ContentType.BINARY.

perform_upload(*local_file_path: str*, *remote_file_path: str*, *content_type:* ContentType = *ContentType.TEXT*, *upload_in_encoding: str = 'utf-8'*) → None

Upload contents of a given file and save it to a file at the given USS path.

Parameters

- **local_file_path** (*str*) – Name of the file to be uploaded
- **remote_file_path** (*str*) – Path of the file where it will be created
- **content_type** (ContentType, *optional*) – Specifies the content type to fetch ("text" or "binary", "text" by default)
- **upload_in_encoding** (*str*) – Specifies encoding schema of the uploaded file

Raises

- ***FileNotFound*** – Thrown when specific file is not found.

- **ValueError** – Content type must be either ContenType.TEXT or ContentType.BINARY.

retrieve_content(*file_path: str*, *content_type:* ContentType = *ContentType.TEXT*, *remote_file_encoding: str = 'IBM-1047'*, *receive_in_encoding: str = 'ISO8859-1'*, *as_stream: bool = False*) → str | None | Response

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

Parameters

- **file_path** (*str*) – Path of the file
- **content_type** (ContentType, *optional*) – The content type to receive ("text" or "binary", "text" by default)
- **remote_file_encoding** (*str, optional*) – Encoding file content originally in (to convert from; by default, it is always being converted from "IBM-1047")
- **receive_in_encoding** (*str, optional*) – Encoding to convert file content to (to convert to; by default, it is always being converted to "ISO8859-1")
- **as_stream** (*bool, optional*) – Specifies whether the response is streamed. Default: False

Returns

Contents of a given USS file in string, or None if the file is empty, or a Response object with content of the file if *as_stream == True*

Return type

Union[str, None, Response]

Raises

ValueError – Content type must be either ContenType.TEXT or ContentType.BINARY.

upload(*input_file: str*, *filepath_name: str*, *encoding: str = 'utf-8'*, *binary: bool = False*) → None

Use *perform_upload()* instead of this deprecated function.

write(*filepath_name: str*, *data: str | bytes*, *encoding: str = 'utf-8'*) → None

Write content to a UNIX file or create it with the content if it does not exist.

Parameters

- **filepath_name** (*str*) – Path of the file
- **data** (*Union[str, bytes]*) – Contents to be written
- **encoding** (*str*) – Specifies the encoding name (e.g. IBM-1047)

Raises

ValueError – Data must be either a string or bytes.

Constants classes

ContentType

class zos_files.zowe.zos_files_for_zowe_sdk.constants.**ContentType**(**values*)

Represents a content type to fetch or upload.

FileType

class zos_files.zowe.zos_files_for_zowe_sdk.constants.**FileType**(**values*)

Class used to represent type of files.

Datasets classes

DatasetOption

class zos_files.zowe.zos_files_for_zowe_sdk.datasets.**DatasetOption**(*like: str | None = None*, *volser: str | None = None*, *unit: str | None = None*, *dsorg: str | None = None*, *alcunit: str | None = None*, *primary: int | None = None*, *secondary: int | None = None*, *dirblk: int | None = None*, *avgblk: int | None = None*, *recfm: str | None = None*, *blksize: int | None = None*, *lrecl: int | None = None*, *storclass: str | None = None*, *mgmtclass: str | None = None*, *dataclass: str | None = None*, *dsntype: str | None = None*)

A dataclass that represents options for creating a dataset.

Parameters

- **like** (*Optional[str]*) – The dataset name to copy attributes from
- **volser** (*Optional[str]*) – The volume serial number that identifies where the dataset resides or should be allocated
- **unit** (*Optional[str]*) – Specifies the type of device on which the dataset is to be stored
- **dsorg** (*Optional[str]*) – Defines the organization of the dataset (PS for sequential, PO for partitioned, DA for direct access)
- **alcunit** (*Optional[str]*) – Specifies the unit of space allocation for the dataset (CYL for cylinders, TRK for tracks, BLK for blocks)
- **primary** (*Optional[int]*) – The amount of primary space to allocate for the dataset
- **secondary** (*Optional[int]*) – The amount of secondary space to allocate if the primary space is exhausted
- **dirblk** (*Optional[int]*) – The number of directory blocks to allocate for a partitioned dataset
- **avgblk** (*Optional[int]*) – The average block size for the dataset
- **recfm** (*Optional[str]*) – The format of the records in the dataset
- **blksize** (*Optional[int]*) – The physical block size used for the dataset
- **lrecl** (*Optional[int]*) – The length of the logical records in the dataset
- **storclass** (*Optional[str]*) – Specifies the storage class to be used for the dataset

- **mgmtclass** (*Optional[str]*) – Specifies the management class
- **dataclass** (*Optional[str]*) – Specifies the data class for the dataset
- **dsntype** (*Optional[str]*) – Specifies the type of dataset

property alcunit: str | None

Get the unit of space allocation.

property avgblk: int | None

Get the average block size.

property blksize: int | None

Get the physical block size.

property dataclass: str | None

Get the data class.

property dirblk: int | None

Get the number of directory blocks.

property dsntype: str | None

Get the type of dataset.

property dsorg: str | None

Get the organization of the dataset.

property like: str | None

Get the dataset name to copy attributes from.

property lrecl: int | None

Get the length of logical records.

property mgmtclass: str | None

Get the management class.

property primary: int | None

Get the primary space allocation.

property recfm: str | None

Get the record format.

property secondary: int | None

Get the secondary space allocation.

property storclass: str | None

Get the storage class.

to_dict() → dict[str, Any]

Return the DatasetOption as a dict.

property unit: str | None

Get the type of device.

property volser: str | None

Get the volume serial number.

Datasets

class zos_files.zowe.zos_files_for_zowe_sdk.datasets.**Datasets**(*connection: dict[str, Any]*, *log: bool = True*)

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

It includes all operations related to datasets.

Parameters

- **connection** (*dict[str, Any]*) – A profile for connection in dict (json) format
- **log** (*bool*) – Flag to disable logger

copy_data_set_or_member(*from_dataset_name: str*, *to_dataset_name: str*, *from_member_name: str | None = None*, *volser: str | None = None*, *alias: bool | None = None*, *to_member_name: str | None = None*, *enq: str | None = None*, *replace: bool = False*) → None

Copy a dataset or member to another dataset or member.

Parameters

- **from_dataset_name** (*str*) – Name of the dataset to copy from
- **to_dataset_name** (*str*) – Name of the dataset to copy to
- **from_member_name** (*Optional[str]*) – Name of the member to copy from
- **volser** (*Optional[str]*) – Volume serial number of the dataset to copy from
- **alias** (*Optional[bool]*) – Alias of the dataset to copy from
- **to_member_name** (*Optional[str]*) – Name of the member to copy to
- **enq** (*Optional[str]*) – Enqueue type for the dataset to copy from
- **replace** (*bool*) – If true, members in the target data set are replaced

Raises

ValueError – Thrown when enq has an invalid value

copy_uss_to_data_set(*from_filename: str*, *to_dataset_name: str*, *to_member_name: str | None = None*, *type: FileType = FileType.TEXT*, *replace: bool = False*) → None

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

create(*dataset_name: str*, *options: DatasetOption | None = None*) → None

Create a sequential or partitioned dataset.

Parameters

- **dataset_name** (*str*) – Name of the dataset to be created
- **options** (*Optional[DatasetOption]*) – A DatasetOption class with property options of the dataset

Raises
ValueError – Thrown when a parameter has an invalid value

create_default(*dataset_name: str*, *default_type: str*) → None

Create a dataset with default options set.

Default options depend on the requested type.

Parameters

- **dataset_name** (*str*) – The name of the dataset
- **default_type** (*str*) – The type of the dataset: "partitioned", "sequential", "classic", "c" or "binary"

Raises
ValueError – Thrown when a parameter is invalid

delete(*dataset_name: str*, *volume: str | None = None*, *member_name: str | None = None*) → None

Delete a sequential or partitioned data.

Parameters

- **dataset_name** (*str*) – The name of the dataset
- **volume** (*Optional[str]*) – The optional volume serial number
- **member_name** (*Optional[str]*) – The name of the member to be deleted

delete_migrated(*dataset_name: str*, *purge: bool = False*, *wait: bool = False*) → None

Delete 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.

download(*dataset_name: str*, *output_file: str*) → None

Use *perform_download(content_type=ContenType.TEXT)* instead of this deprecated function.

download_binary(*dataset_name: str*, *output_file: str*, *with_prefixes: bool = False*) → None

Use *perform_download()* instead of this deprecated function.

get_binary_content(*dataset_name: str*, *stream: bool = False*, *with_prefixes: bool = False*) → bytes | Response

Use *retrieve_content(content_type=ContentType.BINARY)* instead of this deprecated function.

get_content(*dataset_name: str*, *stream: bool = False*) → str | None | Response

Use *retrieve_content()* instead of this deprecated function.

list(*name_pattern: str*, *return_attributes: bool = False*) → DatasetListResponse

Retrieve a list of datasets based on a given pattern.

Parameters

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

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

Return type
DatasetListResponse

list_members(*dataset_name: str*, *member_pattern: str | None = None*, *member_start: str | None = None*, *limit: int = 1000*, *attributes: str = 'member'*) → MemberListResponse

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

Parameters

- **dataset_name** (*str*) – Name of the dataset
- **member_pattern** (*Optional[str]*) – Filters members by name pattern
- **member_start** (*Optional[str]*) – The starting point for listing members
- **limit** (*int*) – The maximum number of members returned
- **attributes** (*str*) – The member attributes to retrieve

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

Return type
MemberListResponse

migrate(*dataset_name: str*, *wait: bool = False*) → None

Migrate 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.

perform_download(*dataset_name: str*, *local_file_path: str*, *content_type:* ContentType = *ContentType.TEXT*) → None

Retrieve the contents of a data set and save it to a local file.

Parameters

- **dataset_name** (*str*) – Name of the dataset to be downloaded
- **local_file_path** (*str*) – Name of the file to be saved locally
- **content_type** (ContentType, *optional*) – The content type to receive (“text”, “binary” or “record” (include a 4 byte big endian record len prefix), “text” by default)

Raises
TypeError – Thrown when the *retrieve_content* request does not return a valid Response object.

perform_upload(*local_file_path: str*, *dataset_name: str*, *content_type:* ContentType = *ContentType.TEXT*, *upload_in_encoding: str = 'utf-8'*) → None

Upload contents of a local file to a data set.

Parameters

- **local_file_path** (*str*) – Name of the file to be uploaded
- **dataset_name** (*str*) – Name of the dataset to be created

- **content_type** (ContentType, *optional*) – The content type to receive ("text", "binary" or "record" (include a 4 byte big endian record len prefix), "text" by default)
- **upload_in_encoding** (*str, optional*) – Specifies the encoding to upload the content in (e.g. IBM-1047, "utf-8" by default)

Raises

FileNotFound – Thrown when a file is not found at provided location

recall_migrated(*dataset_name: str*, *wait: bool = False*) → None

Recall 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

rename(*before_dataset_name: str*, *after_dataset_name: str*) → None

Rename the data set.

Parameters

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

rename_member(*dataset_name: str*, *before_member_name: str*, *after_member_name: str*, *enq: str = ''*) → None

Rename 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.

Raises

ValueError – Thrown when a parameter is invalid

retrieve_content(*dataset_name: str*, *content_type:* ContentType *= ContentType.TEXT*, *as_stream: bool = False*) → str | None | Response

Retrieve the contents of a given dataset.

Parameters

- **dataset_name** (*str*) – The name of the dataset
- **content_type** (ContentType, *optional*) – The content type to receive ("text", "binary" or "record" (include a 4 byte big endian record len prefix), "text" by default)
- **as_stream** (*bool, optional*) – Specifies whether the response is streamed. Default: False

Returns

Contents of a given dataset in string, or None if the dataset is empty, or a Response object with content of the file if *as_stream == True*

Return type
Union[str, None, Response]

upload_file(*input_file: str*, *dataset_name: str*, *encoding: str = 'utf-8'*, *binary: bool = False*) → None
Use *perform_upload()* instead of this deprecated function.

write(*dataset_name: str*, *data: str | bytes*, *encoding: str = 'utf-8'*) → None
Write content to an existing dataset.

Parameters
- **dataset_name** (*str*) – Name of the dataset to retrieve
- **data** (*Union[str, bytes]*) – Content to be written
- **encoding** (*str*) – Specifies encoding name (e.g. IBM-1047) for text data

Raises
ValueError – Data must be either a string or bytes.

Exceptions classes

InvalidPermsOption

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

Parameters
value (*int*) – The value of the permission option

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: dict[str, Any]*, *log: bool = True*)
Class used to represent the base z/OSMF Jobs API.

It includes all operations related to datasets.

Parameters
- **connection** (*dict[str, Any]*) – A profile for connection in dict (json) format

- **log** (*bool*) – Flag to disable logger

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

Cancel 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"

Raises

ValueError – Thrown if the modify_version is invalid

Returns

A JSON object containing the result of the request execution

Return type

StatusResponse

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

Change the job class.

Parameters

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

Raises

ValueError – Thrown if the modify_version is invalid

Returns

A JSON object containing the result of the request execution

Return type

StatusResponse

delete_job(*jobname: str*, *jobid: str*, *modify_version: str = '2.0'*) → StatusResponse

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"

Raises

ValueError – Thrown if the modify_version is invalid

Returns

A JSON object containing the result of the request execution

Return type
StatusResponse

`get_jcl_text`(*correlator: str*) → str

Retrieve the input JCL text for job with specified correlator.

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

Returns
A str object containing the result of the request execution

Return type
str

`get_job_output_as_files`(*status: dict[str, Any]*, *output_dir: str*) → None

Get all spool files and 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** (*dict[str, Any]*) – 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

`get_job_status`(*jobname: str*, *jobid: str*) → JobResponse

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
JobResponse

`get_spool_file_contents`(*correlator: str*, *id: str*) → str

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
The contents of the spool file

Return type
str

get_spool_files(*correlator: str*) → list[SpoolResponse]

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
list[SpoolResponse]

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

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"

Raises
ValueError – Thrown if the modify_version is invalid

Returns
A JSON object containing the result of the request execution

Return type
StatusResponse

list_jobs(*owner: str | None = None*, *prefix: str = '*'*, *max_jobs: int = 1000*, *user_correlator: str | None = None*) → list[JobResponse]

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

Parameters

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

Returns
A list of jobs on JES queue based on the given parameters

Return type
list[JobResponse]

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

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"

Raises
ValueError – Thrown if the modify_version is invalid

Returns
A JSON object containing the result of the request execution

Return type
StatusResponse

submit_from_local_file(*jcl_path: str*) → JobResponse

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
JobResponse

submit_from_mainframe(*jcl_path: str*) → JobResponse

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
JobResponse

submit_plaintext(*jcl: str*) → JobResponse

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
JobResponse

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: dict[str, Any]*, *tso_profile: dict[str, Any] | None = None*, *log: bool = True*)

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

Parameters

- **connection** (*dict[str, Any]*) – Connection object
- **tso_profile** (*Optional[dict[str, Any]]*) – Profile used for tso connection
- **log** (*bool*) – Flag to disable logger

end(*session_key: str*) → EndResponse

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
EndResponse

end_tso_session(*session_key: str*) → str

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: str*) → IssueResponse

Issue 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
IssueResponse

parse_message_ids(*response_json: dict[str, Any]*) → list[str]

Parse TSO response and retrieve only the message ids.

Parameters
response_json (*dict[str, Any]*) – The JSON containing the TSO response

Returns
A list containing the TSO response message ids

Return type
list[str]

ping_tso_session(*session_key: str*) → str

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: list[dict[str, Any]]*) → list[str]

Parse the TSO response and retrieve all messages.

Parameters
response_json (*list[dict[str, Any]]*) – The JSON containing the TSO response

Returns
A list containing the TSO response messages

Return type
list[str]

send(*session_key: str*, *message: str*, *read_reply: bool = True*) → SendResponse

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
- **read_reply** (*bool*) – Whether to read the reply from the TSO session

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

Return type
SendResponse

send_tso_message(*session_key: str*, *message: str*) → list[dict[str, Any]]

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[dict[str, Any]]

start(*proc: str | None = None*, *chset: str | None = None*, *cpage: str | None = None*, *rows: str | None = None*, *cols: str | None = None*, *rsize: str | None = None*, *acct: str | None = None*) → StartResponse

Start a TSO session.

Parameters

- **proc** (*Optional[str]*) – Proc parameter for the TSO session (default is "IZUFPROC")
- **chset** (*Optional[str]*) – Chset parameter for the TSO session (default is "697")
- **cpage** (*Optional[str]*) – Cpage parameter for the TSO session (default is "1047")
- **rows** (*Optional[str]*) – Rows parameter for the TSO session (default is "204")
- **cols** (*Optional[str]*) – Cols parameter for the TSO session (default is "160")
- **rsize** (*Optional[str]*) – Rsize parameter for the TSO session (default is "4096")
- **acct** (*Optional[str]*) – Acct parameter for the TSO session (default is "DEFAULT")

Returns
The 'servletKey' key for the created session (if successful)

Return type
StartResponse

start_tso_session(*proc: str | None = None*, *chset: str | None = None*, *cpage: str | None = None*, *rows: str | None = None*, *cols: str | None = None*, *rsize: str | None = None*, *acct: str | None = None*) → str

Start a TSO session.

Parameters

- **proc** (*Optional[str]*) – Proc parameter for the TSO session (default is "IZUFPROC")
- **chset** (*Optional[str]*) – Chset parameter for the TSO session (default is "697")
- **cpage** (*Optional[str]*) – Cpage parameter for the TSO session (default is "1047")
- **rows** (*Optional[str]*) – Rows parameter for the TSO session (default is "204")
- **cols** (*Optional[str]*) – Cols parameter for the TSO session (default is "160")
- **rsize** (*Optional[str]*) – Rsize parameter for the TSO session (default is "4096")
- **acct** (*Optional[str]*) – 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: dict[str, Any]*, *log: bool = True*)

Class used to represent the base z/OSMF API.

Parameters

- **connection** (*dict[str, Any]*) – The z/OSMF connection object (generated by the ZoweSDK object)
- **log** (*bool*) – Flag to disable logger

get_info() → ZosmfResponse

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
ZosmfResponse

list_systems() → ZosmfResponse

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

Return type
ZosmfResponse

CHAPTER

FOUR

CLASSES

4.1 zowe.secrets_for_zowe_sdk

4.2 zowe.workflows_for_zowe_sdk

4.2.1 Workflows

class workflows.zowe.workflows_for_zowe_sdk.**Workflows**(*connection: dict[str, Any]*, *version: str = '1.0'*)

Representation of the base z/OSMF Workflows API.

See more at https://www.ibm.com/docs/en/zos/3.1.0?topic=services-zosmf-workflow.

Parameters

- **connection** (*dict[str, Any]*) – The connection object
- **version** (*str*) – The supported version of z/OSMF Workflows (1.0 is the only version available for now)

archive_workflow(*workflow_key: str*) → str

You can use this operation to archive a z/OSMF workflow instance on a z/OS system.

https://www.ibm.com/docs/en/zos/3.1.0?topic=services-archive-workflow-instance Query parameters definition: https://www.ibm.com/docs/en/zos/3.1.0?topic=services-retrieve-workflow-definition#GETMethodRetrieveWorkflowDefinition__QueryParameters__title__1

Parameters
workflow_key (*str*) – Identifies the workflow to be archived.

Returns
workflowKey – The archived workflow key.

Return type
str

cancel_workflow(*workflow_key: str*) → str

You can use this operation to cancel a z/OSMF workflow on a z/OS system.

https://www.ibm.com/docs/en/zos/3.1.0?topic=services-cancel-workflow

Parameters
workflow_key (*str*) – Identifies the workflow to be canceled.

Returns
workflowName – The new name of the canceled workflow on successful cancellation.

Return type
str

create_workflow(*workflow_name: str*, *workflow_definition_file: str*, *system: str*, *owner: str*, *workflow_definition_file_system: str | None = None*, *variable_input_file: str | None = None*, *variables: list[dict] | None = None*, *resolve_global_conflict_by_using: Literal['global', 'input'] = 'global'*, *workflow_archive_safid: str | None = None*, *comments: str | None = None*, *assign_to_owner: bool = True*, *access_type: Literal['Public', 'Restricted', 'Private'] = 'Public'*, *account_info: str | None = None*, *job_statement: str | None = None*, *delete_completed_jobs: bool = False*, *jobs_output_directory: str | None = None*, *auto_delete_on_completion: bool = False*, *target_systemuid: str | None = None*, *target_systempwd: str | None = None*) → CreateWorkflowResponse

You can use this operation to create a z/OSMF workflow on a z/OS system.

See https://www.ibm.com/docs/en/zos/3.1.0?topic=services-create-workflow for more information. Detailed request body parameters description: https://www.ibm.com/docs/en/zos/3.1.0?topic=services-create-workflow#POSTMethodCreateAWorkflow__WorkflowRequestContent__title__1

Parameters

- **workflow_name** (*str*) – Descriptive name for the workflow (up to 100 characters).
- **workflow_definition_file** (*str*) – Location of the workflow definition file.
- **system** (*str*) – Nickname of the system on which the workflow is to be created.
- **owner** (*str*) – User ID of the workflow owner.
- **workflow_definition_file_system** (*Optional[str]*) – Nickname of the system on which the specified workflow definition file and any related files reside.
- **variable_input_file** (*Optional[str]*) – Specifies an optional properties file that you can use to pre-specify values for one or more of the variables that are defined in the workflow definition file.
- **variables** (*Optional[list[dict]]*) – A list of one or more variables for this workflow. Empty list by default.
- **resolve_global_conflict_by_using** (*Literal['global', 'input']*) – **Optional** When input variables are provided, this property specifies which type of the variable is used. “global” by default.
- **workflow_archive_safid** (*Optional[str]*) – Indicates who can access the archived workflow, which is archived from the current workflow to a user specified directory. The default value is the current user ID of the workflow owner.
- **comments** (*Optional[str]*) – Specifies any information that you want to associate with the creation of this workflow (up to 500 characters).
- **assign_to_owner** (*bool*) – **Optional** Indicates whether the workflow steps are assigned to the workflow owner when the workflow is created. The default is “true”.
- **access_type** (*Literal['Public', 'Restricted', 'Private']*) – **Optional** Specifies the access type for the workflow. If you omit this property, the workflow is “public”, by default.
- **account_info** (*Optional[str]*) – For a workflow that submits a job, this property specifies the account information to use in the JCL JOB statement.
- **job_statement** (*Optional[str]*) – For a workflow that submits a job, this property specifies the JOB statement JCL that is used in the job.

- **delete_completed_jobs** (*bool*) – **Optional** For a workflow that submits a job, this property specifies whether the job is deleted from the JES spool after it completes successfully. If you omit this property, the completed job is retained on the JES spool.
- **jobs_output_directory** (*Optional[str]*) – For a workflow that submits a job, this property specifies the name of a UNIX directory that is to be used for automatically saving job spool files from the workflow. If you omit this property, the job spool files are not saved.
- **auto_delete_on_completion** (*bool*) – **Optional** Indicates whether the workflow is automatically deleted from the local system when all of its steps are marked complete or skipped. If you omit this property, the workflow instance is retained.
- **target_systemuid** (*Optional[str]*) – The user ID to be used for remote system basic authentication.
- **target_systempwd** (*Optional[str]*) – The password to be used for remote system basic authentication.

Returns
A CreateWorkflowResponse object containing the result created workflow information

Return type
CreateWorkflowResponse

delete_archived_workflow(*workflow_key: str*)

You can use this operation to remove an archived z/OSMF workflow from a z/OS system.

https://www.ibm.com/docs/en/zos/3.1.0?topic=services-delete-archived-workflow

Parameters
workflow_key (*str*) – Identifies the archived workflow to be deleted.

delete_workflow(*workflow_key: str*)

You can use this operation to remove a z/OSMF workflow from a z/OS system.

https://www.ibm.com/docs/en/zos/3.1.0?topic=services-delete-workflow

Parameters
workflow_key (*str*) – Identifies the workflow to be deleted.

get_archived_workflow_properties(*workflow_key: str*, *return_steps_data: bool = False*, *return_variables_data: bool = False*) → GetArchivedWorkflowPropertiesResponse

You can use this operation to retrieve the properties of an archived z/OSMF workflow.

https://www.ibm.com/docs/en/zos/3.1.0?topic=services-get-properties-archived-workflow
Query parameters definition: https://www.ibm.com/docs/en/zos/3.1.0?topic=services-get-properties-archived-workflow#GETMethodRetrieveInformationArchived__QueryParameterReturnData__title__1

Parameters

- **workflow_key** (*str*) – Identifies the archived workflow to be queried.
- **return_steps_data** (*bool*) – **Optional** Response will contain an array of **ArchivedWorkflowStepResponse** objects if True.
- **return_variables_data** (*bool*) – **Optional** Response will contain an array of **ArchivedWorkflowVariableResponse** objects if True.

Returns
A GetArchivedWorkflowPropertiesResponse object the containing archived workflow properties

Return type
GetArchivedWorkflowPropertiesResponse

`get_workflow_definition`(*definition_file_path: str*, *workflow_definition_file_system: str | None = None*, *return_steps_data: bool = False*, *return_variables_data: bool = False*) → GetWorkflowDefinitionResponse

You can use this operation to retrieve the contents of a z/OSMF workflow definition from a z/OS system.

https://www.ibm.com/docs/en/zos/3.1.0?topic=services-retrieve-workflow-definition Query parameters definition: https://www.ibm.com/docs/en/zos/3.1.0?topic=services-retrieve-workflow-definition#GETMethodRetrieveWorkflowDefinition__QueryParameters__title__1

Parameters

- **definition_file_path** (*str*) – Specifies the location of the workflow definition file, which is either a UNIX path name (including the file name) or a fully qualified z/OS data set name.
- **workflow_definition_file_system** (*Optional[str]*) – Nickname of the system on which the specified workflow definition file and any related files reside.
- **return_steps_data** (*bool*) – **Optional** Response will contain an array of **WorkflowDefinitionStepResponse** objects if True.
- **return_variables_data** (*bool*) – **Optional** Response will contain an array of **WorkflowDefinitionVariableResponse** objects if True.

Returns
A GetWorkflowDefinitionResponse object containing the workflow definition file information

Return type
GetWorkflowDefinitionResponse

`get_workflow_properties`(*workflow_key: str*, *return_steps_data: bool = False*, *return_variables_data: bool = False*) → GetWorkflowPropertiesResponse

You can use this operation to retrieve the properties of a z/OSMF workflow.

See https://www.ibm.com/docs/en/zos/3.1.0?topic=services-get-properties-workflow for more information. Query parameters definition: https://www.ibm.com/docs/en/zos/3.1.0?topic=services-get-properties-workflow#GETMethodRetrieveInformationAboutWF__QueryParameterReturnData__title__1

Parameters

- **workflow_key** (*str*) – Identifies the workflow to be queried.
- **return_steps_data** (*bool*) – **Optional** Response will contain an array of **WorkflowStepResponse** objects if True.
- **return_variables_data** (*bool*) – **Optional** Response will contain an array of **WorkflowVariableResponse** objects if True.

Returns
A GetWorkflowPropertiesResponse object containing the workflow properties

Return type
GetWorkflowPropertiesResponse

list_archived_workflows(*order_by: Literal['desc', 'asc'] | None = None*, *view: Literal['user', 'domain'] | None = None*) → list[ListArchivedWorkflowsResponse]

You can use this operation to list the archived z/OSMF workflows for a system or sysplex.

https://www.ibm.com/docs/en/zos/3.1.0?topic=services-list-archived-workflows-system Query parameters definition: https://www.ibm.com/docs/en/zos/3.1.0?topic=services-list-archived-workflows-system#GETMethodListArchivedWorkflows__QueryParametersFilters__title__1

Parameters

- **order_by** (*Optional[Literal['desc', 'asc']]*) – To sort the returned instances by time.
- **view** (*Optional[Literal['user', 'domain']]*) – To select the list instances by view.

Returns

An array of ListArchivedWorkflowsResponse objects containing archived workflows information

Return type

list[ListArchivedWorkflowsResponse]

list_workflows(*workflow_name: str | None = None*, *category: Literal['general', 'configuration'] | None = None*, *system: str | None = None*, *status_name: Literal['in-progress', 'complete', 'automation-in-progress', 'canceled'] | None = None*, *owner: str | None = None*, *vendor: str | None = None*) → list[ListWorkflowsResponse]

You can use this operation to list the z/OSMF workflows for a system or sysplex.

https://www.ibm.com/docs/en/zos/3.1.0?topic=services-list-workflows-system-sysplex Query parameters definition: https://www.ibm.com/docs/en/zos/3.1.0?topic=services-list-workflows-system-sysplex#GETMethodListWorkflows__QueryParametersFilters__title__1

Parameters

- **workflow_name** (*Optional[str]*) – Workflow name. You can specify a regular expression here to match desired workflow names.
- **category** (*Optional[Literal['general', 'configuration']]*) – Category of the workflow, which is either general or configuration.
- **system** (*Optional[str]*) – Nickname of the system on which the workflow is to be performed.
- **status_name** (*Optional[Literal['in-progress', 'complete', 'automation-in-progress', 'canceled']]*) – Workflow status.
- **owner** (*Optional[str]*) – Workflow owner (a valid z/OS user ID).
- **vendor** (*Optional[str]*) – Name of the vendor that provided the workflow definition file.

Returns

A ListWorkflowsResponse array of objects containing the workflows short information

Return type

list[ListWorkflowsResponse]

start_workflow(*workflow_key: str*, *resolve_conflict_by_using: Literal['outputFileValue', 'existingValue', 'leaveConflict'] = 'outputFileValue'*, *step_name: str | None = None*, *perform_subsequent: bool = True*, *notification_url: str | None = None*, *target_systemuid: str | None = None*, *target_systempwd: str | None = None*)

You can use this operation to start a z/OSMF workflow on a z/OS system.

The workflow to be started must contain at least one automated step. https://www.ibm.com/docs/en/zos/3.1.0?topic=services-start-workflow Detailed request body parameters description: https://www.ibm.com/docs/en/zos/3.1.0?topic=services-start-workflow#PUTMethodStartAWorkflow__WorkflowRequestContent__title__1

Parameters

- **workflow_key** (*str*) – Identifies the workflow to be started.
- **resolve_conflict_by_using** (*Literal['outputFileValue', 'existingValue', 'leaveConflict']*) – **Optional** Indicates how variable conflicts, if any, are to be handled when the Workflows task reads in the output file from a step that runs a REXX exec or UNIX shell script.
- **step_name** (*Optional[str]*) – The name of the step at which automation is to begin.
- **perform_subsequent** (*bool*) – **Optional** If the workflow contains any subsequent automated steps, this property indicates whether z/OSMF is to perform the steps.
- **notification_url** (*Optional[str]*) – A notification URL (up to 2000 characters).
- **target_systemuid** (*Optional[str]*) – The user ID to be used for remote system basic authentication.
- **target_systempwd** (*Optional[str]*) – The password to be used for remote system basic authentication.

CHAPTER

FIVE

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)

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 neccessary 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

Symbols

A

B

C

D

E

F

M

N

O

P

R

S