pyCSM

dblea00

Dec 18, 2023

CLIENTS DOCUMENTATION

The pyCSM library is a python library designed to make RESTAPI calls to an IBM Copy Services Manager (CSM) server. CSM is a product used to manage block level replication for IBM Storage Systems.

Solutions in CSM range from point in time and Safeguarded Copy management to more complex two, three or four site replication.

For more details on CSM or for details on the CSM RESTAPI, check out the CSM Documentation.

CHAPTER

ONE

HOW TO USE THE PYCSM LIBRARY

The pyCSM library is broken up into three main sections: Authorization, Clients and Services.

When you use the library you can choose from one of the following configurations.

1.1 Client class only

This option allows the caller to init a python class which manages the authorization automatically for all subsequent calls to the server.

The caller can pass in the authentication information once during the init and then calls to methods withing that class will ensure a token is obtained or **automatically renewed**.

It is recommended that you use the client classes so that you do not have to manage the authorization in your code.

Example:

```
sessClient = session_client.sessionClient("localhost", "9559", "csmadmin",
"csm") print(sessClient.get_session_overviews().json())
```

1.2 Authorization + Services

This option allows the caller to manage the authorization calls itself.

You can use the Authorization methods to obtain a token which can then be passed into the services methods.

Using this option gives a caller greater control over when the token is obtained, however it would be up to the caller to handle failures when the token expires.

Example:

```
token = auth.get_token("https://localhost:9559/CSM/web", "csmadmin", "csm")
print(session_service.get_session_overviews("https://localhost:9559/CSM/web",
token).json())
```

1.3 Services only

This option allows the caller to manage the authorization itself through a means other than using the Authorization call provided by pyCSM.

The caller will need to obtain a token still in order to pass into the Services calls.

Example:

```
print(session_service.get_session_overviews("https://localhost:9559/CSM/web",
token).json())
```

1.4 Specifying Properties

Whether using one of the clients, or using the services, there are default properties that can be set for how the calls are made.

Available properties are defined in a Python dictionary. The current options are the following:

- "language" - Defines the language for returned translated results. Default is "en-US".
- "verify" - Set to True to verify the server side certificate. Default is False.
- "cert" - The client certificate. Default is None.

You can query for the current property values from either the client or the service. A dictionary is returned from both.

Example Using Client:

```
sessClient = session_client.sessionClient(server_address, server_port,
username, password) properties = sessClient.get_properties()
```

Example Using Service:

```
session_service.get_properties()
```

To modify the properties call the change_properties() method on either the client or service depending on what interface you're using. Pass in a dictionary containing the values you wish to change.

Example Changing the language to French:

```
my_properties = {"language":  "de"} sessClient.change_properties(my_properties)
```

After changing the properties all calls to the client or service will use the new properties.

NOTE: The verify property defaults to False and the cert property to None. It is highly recommended that you set verify to True and specify a CA cert to use for a secure connection

Example:

```
my_properties = {"verify":  "True", "cert":  ('/certs/localhost.crt', '/certs/
private.key')} sessClient.change_properties(my_properties)
```

CHAPTER

TWO

CLIENTS, AUTHORIZATION AND SERVICES

2.1 Clients

There are three clients that can be used for various actions on the CSM server.

The *Session Client* class is designed to make calls related to managing CSM sessions and replication.

The *Hardware Client* class is designed to make calls for actions pertaining to setting up and managing connections from CSM to the Storage System.

The *System Client* class is designed to make calls pertaining to the server and system configuration.

2.2 Services

The *Hardware* provides methods around managing the hardware connection from CSM to the storage system or retrieving information from the storage system.

The *System* provides methods for configuring the CSM server, creating log packages, and other server level commands.

The *Sessions* provides the methods for managing sessions on the CSM server. This includes creating and running commands to sessions.

The *Copysets* provides the methods for adding, removing and exporting copy sets from a CSM session.

The *Schedule* provides the methods for viewing and running scheduled tasks on the CSM server.

2.3 Authorization

The *Authorization* module contains the method to obtain a token from the CSM server.

NOTE: This does not need to be used if you are using one of the clients.

CHAPTER

THREE

INDEX FOR PYCSM DOCUMENTATION

3.1 Hardware Client

class pyCSM.clients.hardware_client.**hardwareClient**(*server_address*, *server_port*, *username*, *password*)

The hardwareClient class can be used to call various hardware level commands such as adding device connections, removing device connections, getting lists of volumes, etc. By using the hardwareClient class you enter the username and password only when you instantiate the class which will obtain a token to the server that will be used on all calls using the class. In the event that the token expires, the client will automatically handle the error and retrieve a new token prior to retrying the call. The client makes RESTAPI calls to the server and returns the results. For more details on what is returned from a call, see the CSM Documentation for the specific release.

add_device(*device_type*, *device_ip*, *device_username*, *device_password*, *device_port=None*, *second_ip=None*, *second_port=None*, *second_username=None*, *second_password=None*)

Use this method to create a connection from the CSM server to a specified storage system

Parameters

- **device_type** (*str*) – Type of storage device ex. ds8000 or svc.
- **device_ip** (*str*) – IP address or hostname for the primary HMC for the storage system.
- **device_username** (*str*) – Username for the storage system connection.
- **device_password** (*str*) – Password for the storage system connection.
- **device_port** (*str) (OPTIONAL*) – Port to use for the connection to the storage system.
- **second_ip** (*str) (OPTIONAL*) – For DS8000 storage systems, the IP address or hostname of a secondary HMC.
- **second_port** (*str) (OPTIONAL*) – Port to use for the connection to the secondary HMC.
- **second_username** (*str) (OPTIONAL*) – Username for the connection to the secondary HMC.
- **second_password** (*str) (OPTIONAL*) – Password for the connection to the secondary HMC.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

add_zos_cert(*file_path*)

This method will add a given cert to zos connection.

Parameters **file_path** (*str*) – path for given certificate file.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

add_zos_device(*device_id*)
This method will add a storage system through the zoshost connection.

Parameters

- **device_id** (*str*) – Storage system name in the format "DS8000:BOX:2107.KXZ91".
- **connection_type** (*str*) – type of connection

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

add_zos_host(*host_ip*, *password*, *username*, *host_port*)
This method will create a zos connection to the current IP

Parameters

- **host_ip** (*str*) – Primary IP address for the zos system.
- **password** (*str*) – Password for the zos system connection
- **username** (*str*) – Username for the zos system connection
- **host_port** (*str*) – Port for the zos system

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

static change_properties(*property_dictionary*)
Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –
- **{"language"** (*ex.*) – "en-UK", "verify":True}

Returns Returns the new properties dictionary.

export_vol_writeio_history(*session_name*, *start_time*, *end_time*)
Exports a summary of the write i/o history for all volumes in a session to a csv file between the given times.

Parameters

- **session_name** (*str*) – The name of the session.
- **start_time** (*str*) – Start time YYYY-MM-DD.
- **end_time** (*str*) – End time YYYY-MM-DD.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_devices(*device_type*)
Use this call to return the storage system for all storage systems of the passed in type.

Parameters **device_type** (*str*) – Type of storage device ex. ds8000 or svc.

Returns Returns JSON String representing the result of the command.

get_path_on_storage_system(*system_id*)
Query for all logical paths on the given DS8000 storage system.

Parameters **system_id** (*str*) – The id of the storage system to be updated.

Returns JSON String representing the result of the command.

get_paths()
Queries all the logical paths for all DS8000 storage systems connected to the CSM server.

Returns JSON String representing the result of the command.

static get_properties()
Returns a dictionary of the current properties and their values set for the file.

get_svchosts(*device_id*)
Get the hosts defined on the SVC based storage system

Parameters **device_id** (*str*) – The id of the storage system being used. (ex. "FAB3-DEV13")

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_volumes(*system_name*)
Use this method to retrieve all volumes for a given storage system

Parameters **system_name** (*str*) – The name of the storage system.

Returns JSON String representing all the volumes for that storage system.

get_volumes_by_wwn(*wwn_name*)
Return the information for all volumes based on the list of WWNs passed in.

Parameters **wwn_name** (*str*) – The volume wwn you would like to query or a subset of the volume wwn for a volume list

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_zos_candidate()
This method will query for the devices in REST that are attached to the zos system

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_zos_host()
This method will get the information for all zos host connections.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

map_volumes_to_host(*device_id*, *force*, *hostname*, *is_host_cluster*, *volumes*, *scsi=''*)
Use this method to retrieve all volumes for a given storage system

Parameters

- **device_id** (*str*) – The id for the storage device. (ex. "FAB3-DEV13")
- **force** (*bool*) – boolean of whether user would like to force command
- **hostname** (*str*) – name of the host
- **is_host_cluster** (*bool*) – boolean variable that indicates whether host is a cluster
- **scsi** (*str*) –
- **volumes** (*str) List of volumes to map to the host (ex. ["mVol0_211115100540","mVol1_211115100540"]*) –

Returns JSON String representing all the volumes for that storage system.

refresh_config(*system_id*)
Refreshes the configuration for the given storage system. Issuing this command will force the CSM server to requery the hardware for any new or deleted volumes.

Parameters **system_id** (*str*) – The id of the storage system to be refreshed.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

remove_device(*system_id*)
Use this method to remove the connection to the specified storage system

Parameters **system_id** (*str*) – The id of the storage system to be removed.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

remove_zos_host(*host_ip*, *host_port*)
This method will create a zos connection to the current IP

Parameters

- **host_ip** (*str*) – Primary IP address for the zos system.
- **host_port** (*str*) – Port for the zos system

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

unmap_volumes_to_host(*device_id*, *force*, *hostname*, *is_host_cluster*, *volumes*)
Use this method to retrieve all volumes for a given storage system

Parameters

- **device_id** (*str*) – The id for the storage device (ex. "FAB3-DEV13")
- **force** (*bool*) – boolean of whether user would like to force command
- **hostname** (*str*) – name of the host
- **is_host_cluster** (*bool*) – boolean variable that indicates whether host is a cluster
- **volumes** (*str) List of volumes to map to the host (ex. ["mVol0_211115100540","mVol1_211115100540"]*) –

Returns JSON String representing all the volumes for that storage system.

update_connection_info(*device_ip*, *device_password*, *device_username*, *connection_name*)
Update the userid/pw for a given storage system

Parameters

- **device_ip** (*str*) – Primary IP address for the storage system.
- **device_password** (*str*) – New password for the storage system connection
- **device_username** (*str*) – New user name for the storage system connection
- **connection_name** (*str*) – Name of the connection. ex. HMC:9.11.114.59

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

update_device_site_location(*system_id*, *location*)
Set a user defined site location for a given storage system

Parameters

- **system_id** (*str*) – The id of the storage system to be updated.
- **location** (*str*) – The name of the location to set on the storage system.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

3.2 Session Client

class pyCSM.clients.session_client.**sessionClient**(*server_address*, *server_port*, *username*, *password*)

The sessionClient class can be used to call various session level commands such as creating sessions, adding or removing copy sets, running commands against the session or scheduled task, etc. By using the sessionClient class you enter the username and password only when you instantiate the class which will obtain a token to the server that will be used on all calls using the class. In the event that the token expires, the client will automatically handle the error and retrieve a new token prior to retrying the call.

The client makes RESTAPI calls to the server and returns the results. For more details on what is returned from a call, see the CSM Documentation for the specific release.

add_copysets(*name*, *copyset*, *roleorder=None*)

Add copy sets to a given session

Parameters

- **name** (*str*) – The name of the session.
- **copysets** (*list*) – List of copysets to add to a session ex. (Single Copy set with two volumes)

  [["DS8000:2107.GXZ91:VOL:D000","DS8000:2107.GXZ91:VOL:D001"]]

  ex. (Two Copy sets with two volumes each)

  "[[DS8000:1245.KTLM:VOL:0001","DS8000:1245.KTLM:VOL:0101"],
  ["DS8000:2107.GXZ91:VOL:D004","DS8000:2107.GXZ91:VOL:D005"]]"

- **roleorder** (*list*) – Optional list of the role names depicting the order of the volumes passed in on copysets ex. ["H1", "H2"]

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

static change_properties(*property_dictionary*)

Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –
- **{"language"** (*ex.*) – "en-UK", "verify":True}

Returns Returns the new properties dictionary.

`create_scheduled_task`(*json*)
Creates a new task with given task info

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

`create_session`(*name*, *sess_type*, *desc*)
Create a copy services manager session. A session must be created before copy sets can be placed into the session and managed by the server.

Parameters

- **name** (*str*) – The name of the session that will be created.
- **sess_type** – The type of session to create.
- **desc** (*str) (Optional*) – description for the session

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

`create_session_by_volgroup_name`(*volgroup*, *type*, *desc=None*)
Create a copy services manager session and automatically creates a session name and populates the session based on the passed in volume group

Parameters

- **volgroup** (*str*) – The name of the specv volume group that will be created.
- **type** (*str*) – type The type of session to create. Only Spec V Snapshot supports this today. Type is the "shortname" for the copy type returned in the /system/sessiontypes query.
- **desc** (*str*) – description Optional description for the session

Returns JSON String representing the result of the command.

`delete_session`(*name*)
Deletes a copy services manager session. Only inactive sessions can be deleted.

Args: name (str): The name of the session that will be deleted.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

`disable_scheduled_task`(*taskid*)
Disable a scheduled task from running automatically.

Parameters **taskid** (*str*) – ID of the schedule task to enable. Use the get_scheduled_task() command to get the task id

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

`duplicate_scheduled_task`(*taskid*)
Duplicates scheduled task of a given id

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

`enable_scheduled_task`(*taskid*)
Enable a scheduled task to run based off the schedule defined on the task.

Parameters **taskid** (*str*) – ID of the schedule task to enable. Use the get_scheduled_task() command to get the task id

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

enable_scheduled_task_at_time(*task_id*, *start_time*)
Enable the task at the given time

Parameters

- **task_id** (*int*) – ID of the schedule task to enable
- **start_time** (*str*) – Time to enable the task. Format of yyyy-MM-dd'T'HH-mm (ex. "2022-07-04T12-00")

Returns JSON String representing the result of the command. 'I' = successful,'W' = warning, 'E' = error.

export_copysets(*name*, *file_name*)
Exports copysets from given session as a csv file and downloads it to the calling system.

Parameters

- **name** – Name of the session to export copysets for
- **file_name** – Name for the csv file location (ex. ""/Users/myuser/CSM/Export/myexport.csv")

Returns JSON String representing the result of the command.

export_device_writeio_history(*name*, *start_time*, *end_time*)
Export ESE Box History for a session in csv format to a file

Parameters

- **name** (*str*) – The name of the session.
- **start_time** (*str*) – Start time YYYY-MM-DD
- **end_time** (*str*) – End time YYYY-MM-DD

Returns JSON String representing the result of the command.

export_lss_oos_history(*name*, *rolepair*, *start_time*, *end_time*)
Export LSS OOS History for a session in csv format to a file.

Parameters

- **name** (*str*) – The name of the session.
- **rolepair** (*str*) – The role pair name to query
- **start_time** (*str*) – Start time YYYY-MM-DD
- **end_time** (*str*) – End time YYYY-MM-DD

Returns JSON String representing the result of the command.

get_available_commands(*name*)
Returns the list of available commands for a session based on the session's current state

Parameters **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_backup_details(*name*, *role*, *backup_id*)
Gets detailed information for a given backup in a session.

Parameters

- **name** (*str*) – The name of the session.
- **role** – The name of role where the backups reside.
- **backup_id** – The ID of the backup to send to the run command.

Returns JSON String representing the result of the command.

get_copysets(*name*)
Gets all copy sets and their info for a given session.

Parameters **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_pair_info(*name*, *rolepair*)
Get all the pairs for the session in a given role pair.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **rolepair** (*str*) – The name of the role pair to query in the session

Returns JSON String representing the result of the command. 'I' = successful,'W' = warning, 'E' = error.

static get_properties()
Returns a dictionary of the current properties and their values set for the file.

get_recovered_backup_details(*name*, *backup_id*)
Gets the pair information for a specific recovered backup on a specific session

Parameters

- **name** (*str*) – The name of the session.
- **backup_id** (*int*) – the backupid to get the detailed info for

Returns JSON String representing the result of the command.

get_recovered_backups(*name*)
Gets all recovered backups for Spec V Safeguarded Copy session.

Parameters **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command.

get_rolepair_info(*name*, *rolepair*)
Gets a summary for a given role pair in a session.

Parameters

- **name** (*str*) – The name of the session.
- **rolepair** (*str*) – The name of the role pair.

Returns JSON String representing the result of the command.

get_rpo_history(*name*, *rolepair*, *start_time*, *end_time*)
Export ESE Box History for a session in csv format to a file

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **start_time** (*str*) – Start time YYYY-MM-DD (ex. "2020-04-22")
- **end_time** (*str*) – End time YYYY-MM-DD (ex. "2020-04-22")

Returns JSON String representing the result of the command.

get_scheduled_task(*taskid*)
Returns the scheduled task info of a given task id

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_scheduled_tasks()
Returns a list of scheduled tasks defined on the server

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_session_info(*name*)
This method returns the detailed information for a given session.

Parameters **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_session_options(*name*)
Gets the options for the given session. The results returned from this method will vary depending on the session type.

Parameters **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command.

get_session_overviews()
This method returns the overview summary information for all sessions managed by the server

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_session_overviews_short()
This method returns minimal overview summary information for all sessions managed by the server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_snapshot_clone_details_by_name(*name*, *snapshot_name*)
Gets the pair details for the thin clone of the specified snapshot in the session

Parameters

- **name** (*str*) – The name of the session.
- **snapshot_name** (*str*) – the name of the snapshot to get clone details for

Returns JSON String representing the result of the command.

get_snapshot_clones(*name*)
Gets all clones for snapshots in for Spec V Safeguarded Copy session.

Parameters **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command.

get_snapshot_details_by_name(*name*, *role*, *snapshot_name*)
Gets detailed information for a given snapshot in a session.

Parameters

- **name** (*str*) – The name of the session.
- **role** – The name of role where the snapshot resides.
- **snapshot_name** – The name of the snapshot to return

Returns JSON String representing the result of the command.

modify_session_description(*name*, *desc*)
Changes the description field for a given session.

Parameters

- **name** (*str*) – The name of the session.
- **desc** (*str*) – description for the session

Returns JSON String representing the result of the command.

remove_copysets(*name*, *copysets*, *force=None*, *soft=None*)
Removes Copy Sets from the given session.

Parameters

- **name** (*str*) – The name of the session.
- **copyset** (*str*) – List of copy sets to add to the session. ex. "DS8000:1245.KTLM:VOL:0001", "DS8000:1245.KTLM:VOL:0101"
- **force** (*boolean*) – Force Set to true if you wish to remove the pair from CSM ignoring hardware errors.
- **soft** (*boolean*) – Keep base relationships on the hardware but remove the copy set from the session.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

run_backup_command(*name*, *role*, *backup_id*, *cmd*)
Used to perform a recover or expire for the specified backup.

Parameters

- **name** (*str*) – The name of the session.
- **role** – The name of role where the backups reside.
- **backup_id** – The ID of the backup to send to the run command.
- **cmd** (*str*) – command to run (ex. "Recover Backup", "Expire Backup")

Returns JSON String representing the result of the command.

run_scheduled_task(*taskid*, *synchronous=False*)
Run a scheduled task immediately. Synchronous value set to true if call should not return until task is complete. False if you want it to run in the asynchronous after the call completes.

Parameters

- **taskid** (*str*) – ID of the schedule task to enable.
- **synchronous** (*boolean*) – True if you don't want the command to complete until the task completes

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

run_scheduled_task_at_time(*task_id*, *start_time*)
Run a scheduled task immediately.

Parameters

- **task_id** (*int*) – ID of the schedule task to enable
- **start_time** (*str*) – Time to enable the task. Format of yyyy-MM-dd'T'HH-mm (ex. "2022-07-04T12-00")

Returns JSON String representing the result of the command. 'I' = successful,'W' = warning, 'E' = error.

run_session_command(*ses_name*, *com_name*)
Run a command against a session.

Parameters

- **ses_name** (*str*) – The name of the session.
- **com_name** (*str*) – The name of the command.

Returns JSON String representing the result of the command.

sgc_recover(*ses_name*, *com_name*, *role*, *backup_id*)
Run a Recover command to the specified Safeguarded Copy backup ID.

Parameters

- **ses_name** (*str*) – The name of the session.
- **com_name** (*str*) – The name of the command.
- **role** – The name of role where the backups reside.
- **backup_id** – The ID of the backup to send to the run command.

Returns JSON String representing the result of the command.

wait_for_state(*ses_name*, *state*, *minutes*, *debug=False*)
Runs until the session is in a given state or until it times out and returns the results.

Parameters

- **ses_name** (*str*) – The name of the session.
- **state** (*str*) – state of the server that user wants to wait for.
- **minutes** (*double*) – number of minutes before it times out
- **debug** (*boolean*) – True if you want the state and status to print in console

Returns boolean for whether the state was reached and "session_info": JSON string representing the response of the command

Return type A dictionary with "state_reached"

3.3 System Client

class pyCSM.clients.system_client.**systemClient**(*server_address*, *server_port*, *username*, *password*)

The systemClient class can be used to call various server level commands such as creating log packages, backing up the server, setting up active/standby support, etc. By using the systemClient class you enter the username and password only when you instantiate the class which will obtain a token to the server that will be used on all calls using the class. In the event that the token expires, the client will automatically handle the error and retrieve a new token prior to retrying the call.

The client makes RESTAPI calls to the server and returns the results. For more details on what is returned from a call, see the CSM Documentation for the specific release.

approve_dual_control_request(*id*)
Approve a dual control request

Parameters

- **id** (*int*) – ID of the request caller wants to approve.
- **getDualControlEvents.** (*ID from the 'requestid' field return from*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

backup_server()
Creates a zip backup of the CSM server data that can be used for restoring the server at a later date

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

backup_server_and_download(*file_name*)
Create and downloads a server backup.'

Parameters **file_name** – The file to write the server backup to

Returns Server backup data that is written to the specified file.

change_dual_control_state(*enable*)
Use this method to enable or disable dual control on the CSM server.

Parameters

- **enable** (*bool*) – Set to 'true' if you want to enable dual control
- **disable.** (*or 'false' if you want to*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

static change_properties(*property_dictionary*)
Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –

- **{"language"** (*ex.*) – "en-UK", "verify":True}

Returns Returns the new properties dictionary.

create_and_download_log_pkg(*file_name*)
This method will package all log files on the server into a .jar file that can be used for support - this call is a synchronous call and will not return to caller until package is complete. Call make take a while

Parameters **file_name** – Name of the file to write the log package to

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

create_log_pkg()
This method will package all log files on the server into a .jar file

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_active_standby_status()
Get the current state of the active standby server connection

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_dual_control_requests()
Returns a list of dual control events waiting for approval or rejection

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_dual_control_state()
Use this method to determine if dual control is currently enabled of disabled on the server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_log_events(*count*, *session=None*)
get a list of the most recent log events

Parameters

- **count** (*int*) – The number of messages to return
- **session** (*string*) – (optional) filter messages on session

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_log_pkgs()
Gets a list of log packages and their location on the server

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

static get_properties()
Returns a dictionary of the current properties and their values set for the file.

get_server_backups()
Retrieves a list of all server backups.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_server_version()
Get the version of the server being called

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_session_types()
Get supported session types

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

get_volume_counts()
Get a summary of the volume usage on the server

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

reconnect_active_standby_server()
Reconnect the active standby connection

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

reject_dual_control_request(*id*, *comment*)
Reject a dual control request

Parameters

- **id** (*int*) – ID of the request caller wants to approve.
- **getDualControlEvents.** (*ID from the 'requestid' field return from*) –
- **comment** (*str*) – Comment to the creator of the event on why the
- **rejected.** (*request was*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

remove_active_or_standby_server(*ha_server*)
This method removes the alternate CSM server. If issued to the active server the standby will be removed.

Parameters **ha_server** (*str*) – hostname of the server to remove

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

rest_delete(*url*, *data*, *headers*)
Method to call a REST delete call if one is needed but missing from the package.

url (str): url for the csm server and the rest call the user wants to run data (dict): Dictionary of variables used in the body of a REST call.

ex. params = {"type": device_type, "deviceip": device_ip, "deviceport": device_port,}

headers (dict): Dictionary of variables used in the headers of a REST call except for the token

ex. headers = {"Accept-Language": properties["language"], "Content-Type": "application/x-www-form-urlencoded"}

rest_get(*url*, *data*, *headers*)
Method to call a REST get call if one is needed but missing from the package.

url (str): url for the csm server and the rest call the user wants to run data (dict): Dictionary of variables used in the body of a REST call.

ex. params = {"type": device_type, "deviceip": device_ip, "deviceport": device_port,}

headers (dict): Dictionary of variables used in the headers of a REST call except for the token

ex. headers = {"Accept-Language": properties["language"], "Content-Type": "application/x-www-form-urlencoded"}

`rest_post`(*url*, *data*, *headers*)

Method to call a REST post call if one is needed but missing from the package.

url (str): url for the csm server and the rest call the user wants to run data (dict): Dictionary of variables used in the body of a REST call.

ex. params = {"type": device_type, "deviceip": device_ip, "deviceport": device_port,}

headers (dict): Dictionary of variables used in the headers of a REST call except for the token

ex. headers = {"Accept-Language": properties["language"], "Content-Type": "application/x-www-form-urlencoded"}

`rest_put`(*url*, *data*, *headers*)

Method to call a REST put call if one is needed but missing from the package.

url (str): url for the csm server and the rest call the user wants to run data (dict): Dictionary of variables used in the body of a REST call.

ex. params = {"type": device_type, "deviceip": device_ip, "deviceport": device_port,}

headers (dict): Dictionary of variables used in the headers of a REST call except for the token

ex. headers = {"Accept-Language": properties["language"], "Content-Type": "application/x-www-form-urlencoded"}

`set_property`(*file*, *property_name*, *value*)

This call will set the property provided to the value provide in the selected file

Parameters

- **file** (*str*) – One of the following files can be specified. "server", "bootstrap", "essniclient" or "zosclient"
- **property_name** (*str*) – name of the property to set ex. csm.server.extra_driver_debug
- **value** (*str*) – value to set the property to

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

`set_server_as_standby`(*active_server*)

Issue this command to the server that you want to be the standby server. Sets the server passed in to be the active server. All data on the called server will be replaced with the data from the active server.

Parameters

- **active_server** (*str*) – IP or hostname of the active server.
- **port.** (*This method will use the default*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

set_standby_server(*standby_server*, *standby_username*, *standby_password*)
Reconnect the active standby connection

Parameters

- **standby_server** (*str*) – IP or hostname of the standby server
- **standby_username** (*str*) – Username to create a connection to the standby server
- **standby_password** (*str*) – Password for the user to create a connection to the standby server

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

takeover_standby_server()
Issues a takeover on the standby server making the standby server an active server

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

3.4 Copysets

pyCSM.services.session_service.copyset_service.**add_copysets**(*url*, *tk*, *name*, *copysets*, *roleorder=None*)
Add copy sets to a given session

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **copysets** (*list*) – List of copysets to add to a session ex. (Single Copy set with two volumes)

  [["DS8000:2107.GXZ91:VOL:D000","DS8000:2107.GXZ91:VOL:D001"]]

  ex. (Two Copy sets with two volumes each)

  "[[DS8000:1245.KTLM:VOL:0001","DS8000:1245.KTLM:VOL:0101"],
  ["DS8000:2107.GXZ91:VOL:D004","DS8000:2107.GXZ91:VOL:D005"]]"

- **roleorder** (*list*) – Optional list of the role names depicting the order of the volumes passed in on copysets ex. ["H1", "H2"]

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.copyset_service.**change_properties**(*property_dictionary*)
Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –

- **{"language"** (*ex.*) – “en-UK”, “verify”:True}

Returns Returns the new properties dictionary.

pyCSM.services.session_service.copyset_service.**enable_scheduled_task_at_time**(*url*, *tk*, *task_id*, *start_time*)

Enable the task at the given time

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **task_id** (*int*) – ID of the schedule task to enable
- **start_time** (*str*) – Time to enable the task.
- **yyyy-MM-dd'T'HH-mm.** (*Format of*) –

Returns JSON String representing the result of the command. ‘I’ = successful,’W’ = warning, ‘E’ = error.

pyCSM.services.session_service.copyset_service.**export_copysets**(*url*, *tk*, *name*, *file_name*)

Exports copysets as a csv file and downloads it to the calling system.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** – Name of the session to export copysets for
- **file_name** – Name for the csv file location (ex. “”/Users/myuser/CSM/Export/myexport.csv”)

Returns JSON String representing the result of the command.

pyCSM.services.session_service.copyset_service.**get_copysets**(*url*, *tk*, *name*)

Gets all copy sets and their info for a given session.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command. ‘I’ = successful,’W’ = warning, ‘E’ = error.

pyCSM.services.session_service.copyset_service.**get_pair_info**(*url*, *tk*, *name*, *rolepair*)

Get all the pairs for the session in a given role pair.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **rolepair** (*str*) – The name of the role pair to query in the session

Returns JSON String representing the result of the command. ‘I’ = successful,’W’ = warning, ‘E’ = error.

pyCSM.services.session_service.copyset_service.**get_properties**()
Returns a dictionary of the current properties and their values set for the file.

pyCSM.services.session_service.copyset_service.**remove_copysets**(*url*, *tk*, *name*, *copysets*, *force=False*, *soft=False*)
Removes Copy Sets from the given session.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **copyset** (*str*) – list of copyset hosts to remove from a session ex. ["DS8000:2107.GXZ91:VOL:D000"] ex. ["DS8000:1245.KTLM:VOL:0001", "DS8000:2107.GXZ91:VOL:D004"]
- **force** (*boolean*) – Force Set to true if you wish to remove the pair from CSM ignoring hardware errors.
- **soft** (*boolean*) – Keep base relationships on the hardware but remove the copy set from the session.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.copyset_service.**run_scheduled_task_at_time**(*url*, *tk*, *task_id*, *start_time*)
Run a scheduled task immediately.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **task_id** (*int*) – ID of the schedule task to enable
- **start_time** (*str*) – Time to enable the task.
- **yyyy-MM-dd'T'HH-mm.** (*Format of*) –

Returns JSON String representing the result of the command. 'I' = successful,'W' = warning, 'E' = error.

3.5 Schedule

pyCSM.services.session_service.schedule_service.**change_properties**(*property_dictionary*)
Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –
- **{"language"** (*ex.*) – "en-UK", "verify":True}

Returns Returns the new properties dictionary.

pyCSM.services.session_service.schedule_service.**create_scheduled_task**(*url*, *tk*, *json*)
Creates a new task with given task info

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **json** (*str*) – json string of scheduled task info used to create new scheduled task

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.schedule_service.**disable_scheduled_task**(*url*, *tk*, *taskid*)
Disable a scheduled task from running automatically.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **taskid** (*str*) – ID of the schedule task to enable.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.schedule_service.**duplicate_scheduled_task**(*url*, *tk*, *taskid*)
Duplicates scheduled task of a given id

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.schedule_service.**enable_scheduled_task**(*url*, *tk*, *taskid*)
Enable a scheduled task to run based off the schedule defined on the task.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **taskid** (*str*) – ID of the schedule task to enable.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.schedule_service.**get_properties**()
Returns a dictionary of the current properties and their values set for the file.

pyCSM.services.session_service.schedule_service.**get_scheduled_task**(*url*, *tk*, *taskid*)
Returns the scheduled task info of a given task id

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.schedule_service.**get_scheduled_tasks**(*url*, *tk*)
Returns a list of scheduled tasks defined on the server

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.schedule_service.**run_scheduled_task**(*url*, *tk*, *taskid*, *synchronous=False*)

Run a scheduled task immediately. Synchronous value set to true if call should not return until task is complete. False if you want it to run in the asynchronous after the call completes.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **taskid** (*str*) – ID of the schedule task to enable.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

3.6 Sessions

pyCSM.services.session_service.session_service.**change_properties**(*property_dictionary*)

Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –
- **{"language"** (*ex.*) – "en-UK", "verify":True}

Returns Returns the new properties dictionary.

pyCSM.services.session_service.session_service.**create_session**(*url*, *tk*, *name*, *sess_type*, *desc=None*)

Create a copy services manager session. A session must be created before copy sets can be placed into the session and managed by the server.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session that will be created.
- **sess_type** – The type of session to create.
- **desc** (*str) (Optional*) – description for the session

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.session_service.**create_session_by_volgroup_name**(*url*, *tk*, *volgroup*, *type*, *desc=None*)

Create a copy services manager session and automatically creates a session name and populates the session based on the passed in volume group

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **volgroup** (*str*) – The name of the specv volume group that will be created.
- **type** (*str*) – type The type of session to create. Only Spec V Snapshot supports this today. Type is the "shortname" for the copy type returned in the /system/sessiontypes query.
- **desc** (*str*) – description Optional description for the session

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**delete_session**(*url*, *tk*, *name*)

Deletes a copy services manager session. Only inactive sessions can be deleted.

Args: url (str): Base url of CSM server. ex. https://servername:port/CSM/web. tk (str): Rest token for the CSM server. name (str): The name of the session that will be deleted.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.session_service.**export_device_writeio_history**(*url*, *tk*, *name*, *start_time*, *end_time*)

Export ESE Box History for a session in csv format to a file

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **start_time** (*str*) – Start time YYYY-MM-DD
- **end_time** (*str*) – End time YYYY-MM-DD

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**export_lss_oos_history**(*url*, *tk*, *name*, *rolepair*, *start_time*, *end_time*)

Export LSS OOS History for a session in csv format to a file

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **rolepair** (*str*) – The role pair name to query

- **start_time** (*str*) – Start time YYYY-MM-DD
- **end_time** (*str*) – End time YYYY-MM-DD

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_available_commands**(*url*, *tk*, *name*)
Returns the list of available commands for a session based on the session's current state

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.session_service.**get_backup_details**(*url*, *tk*, *name*, *role*, *backup_id*)
Gets detailed information for a given backup in a session.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **role** – The name of role where the backups reside.
- **backup_id** – The ID of the backup to send to the run command.

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_properties**()
Returns a dictionary of the current properties and their values set for the file.

pyCSM.services.session_service.session_service.**get_recovered_backup_details**(*url*, *tk*, *name*, *backup_id*)
Gets the pair information for a specific recovered backup on a specific session

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **backup_id** (*int*) – the backupid to get the detailed info for

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_recovered_backups**(*url*, *tk*, *name*)
Gets all recovered backups for Spec V Safeguarded Copy session.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_rolepair_info**(*url*, *tk*, *name*, *rolepair*)
Gets a summary for a given role pair in a session.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **rolepair** (*str*) – The name of the role pair.

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_rpo_history**(*url*, *tk*, *name*, *rolepair*, *start_time*, *end_time*)
Export ESE Box History for a session in csv format to a file

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **start_time** (*str*) – Start time YYYY-MM-DD (ex. "2020-04-22")
- **end_time** (*str*) – End time YYYY-MM-DD (ex. "2020-04-22")

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_session_info**(*url*, *tk*, *name*)
This method returns the detailed information for a given session.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.session_service.**get_session_options**(*url*, *tk*, *name*)
Gets the options for the given session. The results returned from this method will vary depending on the session type.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_session_overviews**(*url*, *tk*)
This method returns the overview summary information for all sessions managed by the server

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.session_service.**get_session_overviews_short**(*url*, *tk*)
This method returns minimal overview summary information for all sessions managed by the server.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.session_service.session_service.**get_snapshot_clone_details_by_name**(*url*, *tk*, *name*, *snapshot_name*)
Gets the pair details for the thin clone of the specified snapshot in the session

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **snapshot_name** (*str*) – the name of the snapshot to get clone details for

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_snapshot_clones**(*url*, *tk*, *name*)
Gets all clones for snapshots in for Spec V Safeguarded Copy session.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**get_snapshot_details_by_name**(*url*, *tk*, *name*, *role*, *snapshot_name*)
Gets detailed information for a given snapshot in a session.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **role** – The name of role where the snapshot resides.
- **snapshot_name** – The name of the snapshot to return

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**modify_session_description**(*url*, *tk*, *name*, *desc*)
Changes the description field for a given session.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **desc** (*str*) – description for the session

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**run_backup_command**(*url*, *tk*, *name*, *role*, *backup_id*, *cmd*)

Used to perform a recover or expire for the specified backup.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **name** (*str*) – The name of the session.
- **role** – The name of role where the backups reside.
- **backup_id** – The ID of the backup to send to the run command.
- **cmd** (*str*) – command to run (ex. "Recover Backup", "Expire Backup")

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**run_session_command**(*url*, *tk*, *ses_name*, *com_name*)

Run a command against a session.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **ses_name** (*str*) – The name of the session.
- **com_name** (*str*) – The name of the command.

Returns JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**sgc_recover**(*url*, *tk*, *ses_name*, *com_name*, *role*, *backup_id*)

Run a Recover command to the specified Safeguarded Copy backup ID.

Args: url (str): Base url of CSM server. ex. https://servername:port/CSM/web. tk (str): Rest token for the CSM server. ses_name (str): The name of the session. com_name (str): The name of the command. role: The name of role where the backups reside. backup_id: The ID of the backup to send to the run command.

Returns: JSON String representing the result of the command.

pyCSM.services.session_service.session_service.**wait_for_state**(*url*, *tk*, *ses_name*, *state*, *minutes=5*, *debug=False*)

Runs until the session is in a given state or until it times out and returns the results.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

- **ses_name** (*str*) – The name of the session.
- **state** (*str*) – state of the server that user wants to wait for.
- **minutes** (*double*) – number of minutes before it times out
- **debug** (*boolean*) – True if you want the state and status to print in console

Returns boolean for whether the state was reached and "session_info": JSON string representing the response of the command

Return type A dictionary with "state_reached"

3.7 Hardware

pyCSM.services.hardware_service.hardware_service.**add_device**(*url*, *tk*, *device_type*, *device_ip*, *device_username*, *device_password*, *device_port=None*, *second_ip=None*, *second_port=None*, *second_username=None*, *second_password=None*)

Use this method to create a connection from the CSM server to a specified storage system

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **device_type** (*str*) – Type of storage device ex. ds8000 or svc.
- **device_ip** (*str*) – IP address or hostname for the primary HMC for the storage system.
- **device_username** (*str*) – Username for the storage system connection.
- **device_password** (*str*) – Password for the storage system connection.
- **device_port** (*str) (OPTIONAL*) – Port to use for the connection to the storage system.
- **second_ip** (*str) (OPTIONAL*) – For DS8000 storage systems, the IP address or hostname of a secondary HMC.
- **second_port** (*str) (OPTIONAL*) – Port to use for the connection to the secondary HMC.
- **second_username** (*str) (OPTIONAL*) – Username for the connection to the secondary HMC.
- **second_password** (*str) (OPTIONAL*) – Password for the connection to the secondary HMC.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**add_zos_cert**(*url*, *tk*, *file_path*)

This method will add a given cert to zos connection.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **file_path** (*str*) – Path for given certificate file.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**add_zos_device**(*url*, *tk*, *device_id*)
This method will add a storage system through the zoshost connection.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **device_id** (*str*) – Storage system name in the format "DS8000:BOX:2107.KXZ91".

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**add_zos_host**(*url*, *tk*, *host_ip*, *password*, *username*, *host_port*)
This method will create a zos connection to the current IP

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **host_ip** (*str*) – Primary IP address for the zos system.
- **password** (*str*) – Password for the zos system connection
- **username** (*str*) – Username for the zos system connection
- **host_port** (*str*) – Port for the zos system

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**change_properties**(*property_dictionary*)
Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –
- **{"language"** (*ex.*) – "en-UK", "verify":True}

Returns Returns the new properties dictionary.

pyCSM.services.hardware_service.hardware_service.**export_vol_writeio_history**(*url*, *tk*, *session_name*, *start_time*, *end_time*)
Exports a summary of the write i/o history for all volumes in a session to a csv file between the given times.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **session_name** (*str*) – The name of the session.
- **start_time** (*str*) – Start time YYYY-MM-DD. Type:str
- **end_time** (*str*) – End time YYYY-MM-DD.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**get_devices**(*url*, *tk*, *device_type*)
Uses a get request to get info of all the storagedevices of a given type.

Parameters

- **url** (*str*) – Base url of csm server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **device_type** (*str*) – Type of storage device ex. ds8000 or svc.

Returns Returns JSON String representing the result of the command.

pyCSM.services.hardware_service.hardware_service.**get_path_on_storage_system**(*url*, *tk*, *system_id*)
Query for all logical paths on the given DS8000 storage system.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **system_id** (*str*) – The id of the storage system to be updated.

Returns JSON String representing the result of the command.

pyCSM.services.hardware_service.hardware_service.**get_paths**(*url*, *tk*)
Queries all the logical paths for all DS8000 storage systems connected to the CSM server.

Args: url (str): Base url of CSM server. ex. https://servername:port/CSM/web. tk (str): Rest token for the CSM server.

Returns: JSON String representing the result of the command.

pyCSM.services.hardware_service.hardware_service.**get_properties**()
Returns a dictionary of the current properties and their values set for the file.

pyCSM.services.hardware_service.hardware_service.**get_svchosts**(*url*, *tk*, *device_id*)
Get the hosts defined on the SVC based storage system

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **device_id** (*str*) – The id of the storage system being used. (ex. "FAB3-DEV13")

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**get_volumes**(*url*, *tk*, *system_name*)
Use this method to retrieve all volumes for a given storage system

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **system_name** (*str*) – The name of the storage system.

Returns JSON String representing all the volumes for that storage system.

pyCSM.services.hardware_service.hardware_service.**get_volumes_by_wwn**(*url*, *tk*, *wwn_name*)
Return the information for all volumes based on the list of WWNs passed in.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **wwn_name** (*str*) – The volume wwn you would like to query or a subset of the volume wwn for a volume list

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**get_zos_candidate**(*url*, *tk*)
This method will query for the devices in REST that are attached to the zos system.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**get_zos_host**(*url*, *tk*)
This method will get the information for all zos host connections.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**map_volumes_to_host**(*url*, *tk*, *device_id*, *force*, *hostname*, *is_host_cluster*, *volumes*, *scsi=''*)
Map volumes to a host

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **device_id** (*str*) – The id for the storage device (ex. "FAB3-DEV13")
- **force** (*bool*) – boolean of whether user would like to force command
- **hostname** (*str*) – name of the host
- **is_host_cluster** (*bool*) – boolean variable that indicates whether host is a cluster
- **scsi** (*str*) –
- **volumes** (*str) List of volumes to map to the host (ex. ["mVol0_211115100540","mVol1_211115100540"]*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**refresh_config**(*url*, *tk*, *system_id*)

Refreshes the configuration for the given storage system. Issuing this command will force the CSM server to requery the hardware for any new or deleted volumes.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **system_id** (*str*) – The id of the storage system to be updated.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**remove_device**(*url*, *tk*, *system_id*)

Use this method to remove the connection to the specified storage system

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **system_id** (*str*) – The id of the storage system to be removed.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**remove_zos_host**(*url*, *tk*, *host_ip*, *host_port*)

This method will create a zos connection to the current IP

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **host_ip** (*str*) – Primary IP address for the zos system.
- **host_port** (*str*) – Port for the zos system

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**unmap_volumes_to_host**(*url*, *tk*, *device_id*, *force*, *hostname*, *is_host_cluster*, *volumes*)

UnMap volumes from a host

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **device_id** (*str*) – The id for the storage device (ex. "FAB3-DEV13")
- **force** (*bool*) – boolean of whether user would like to force command
- **hostname** (*str*) – name of the host
- **is_host_cluster** (*bool*) – boolean variable that indicates whether host is a cluster

- **volumes** (*str) List of volumes to map to the host (ex. ["mVol0_211115100540","mVol1_211115100540"]*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**update_connection_info**(*url*, *tk*, *device_ip*, *device_password*, *device_username*, *connection_name*)

Update the userid/pw for a given storage system

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **device_ip** (*str*) – Primary IP address for the storage system.
- **device_password** (*str*) – New password for the storage system connection
- **device_username** (*str*) – New user name for the storage system connection
- **connection_name** (*str*) – Name of the connection. ex. HMC:9.11.114.59

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.hardware_service.hardware_service.**update_device_site_location**(*url*, *tk*, *system_id*, *location*)

Set a user defined site location for a given storage system

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **system_id** (*str*) – The id of the storage system to be updated.
- **location** (*str*) – The name of the location to set on the storage system.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

3.8 System

pyCSM.services.system_service.system_service.**approve_dual_control_request**(*url*, *tk*, *id*)

Approve a dual control request

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **id** (*int*) – ID of the request caller wants to approve.
- **getDualControlEvents.** (*ID from the 'requestid' field return from*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**backup_server**(*url*, *tk*)
Creates a zip backup of the CSM server data that can be used for restoring the server at a later date

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**backup_server_and_download**(*url*, *tk*, *file_name*)
Create and downloads a server backup.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **file_name** – The file to write the server backup to

Returns A file downloaded into the client with the specified filename

pyCSM.services.system_service.system_service.**change_dual_control_state**(*url*, *tk*, *enable*)
Use this method to enable or disable dual control on the CSM server.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **enable** (*bool*) – Set to 'true' if you want to enable dual control or
- **disable.** (*'false' if you want to*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**change_properties**(*property_dictionary*)
Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –
- **{"language"** (*ex.*) – "en-UK", "verify":True}

Returns Returns the new properties dictionary.

pyCSM.services.system_service.system_service.**create_and_download_log_pkg**(*url*, *tk*, *file_name*)
This method will package all log files on the server into a .jar file that can be used for support - this call is a synchronous call and will not return to caller until package is complete. Call make take a while

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **file_name** – Name of the file to write the log package to

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**create_log_pkg**(*url*, *tk*)
This method will package all log files on the server into a .jar file

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_active_standby_status**(*url*, *tk*)
Get the current state of the active standby server connection

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_dual_control_requests**(*url*, *tk*)
Returns a list of dual control events waiting for approval or rejection

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_dual_control_state**(*url*, *tk*)
Use this method to determine if dual control is currently enabled of disabled on the server.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_log_events**(*url*, *tk*, *count*, *session=None*)
get a list of the most recent log events

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **count** (*int*) – The number of messages to return
- **session** (*string*) – (optional) filter messages on session

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_log_pkgs**(*url*, *tk*)
Gets a list of log packages and their location on the server

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_properties**()
Returns a dictionary of the current properties and their values set for the file.

pyCSM.services.system_service.system_service.**get_server_backups**(*url*, *tk*)
Retrieves a list of all server backups.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_server_version**(*url*, *tk*)
Get the version of the server being called

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_session_types**(*url*, *tk*)
Get supported session types

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**get_volume_counts**(*url*, *tk*)
Get a summary of the volume usage on the server

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**reconnect_active_standby_server**(*url*, *tk*)
Reconnect the active standby connection

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**reject_dual_control_request**(*url*, *tk*, *id*, *comment*)
Reject a dual control request

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **id** (*int*) – ID of the request caller wants to approve.
- **getDualControlEvents.** (*ID from the 'requestid' field return from*) –
- **comment** (*str*) – Comment to the creator of the event on why the request was rejected.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**remove_active_or_standby_server**(*url*, *tk*, *haServer*)
Remove the alternate server

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **haServer** (*str*) – hostname of the server to remove

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**set_property**(*url*, *tk*, *file*, *property_name*, *value*)
This call will set the property provided to the value provide in the selected file

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **file** (*str*) – One of the following files can be specified. "server", "bootstrap", "essni-client" or "zosclient"
- **property_name** (*str*) – name of the property to set ex. csm.server.extra_driver_debug
- **value** (*str*) – value to set the property to

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**set_server_as_standby**(*url*, *tk*, *active_server*)
Issue this command to the server that you want to be the standby server. Sets the server passed in to be the active server. All data on the called server will be replaced with the data from the active server.

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **active_server** (*str*) – IP or hostname of the active server.
- **port.** (*This method will use the default*) –

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**set_standby_server**(*url*, *tk*, *standby_server*, *standby_username*, *standby_password*)

Sets the server passed in to be the standby server. All data on the passed in server will be replaced with the data from the called server

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.
- **standby_server** (*str*) – IP or hostname of the standby server
- **standby_username** (*str*) – Username to create a connection to the standby server
- **standby_password** (*str*) – Password for the user to create a connection to the standby server

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

pyCSM.services.system_service.system_service.**takeover_standby_server**(*url*, *tk*)

Issues a takeover on the standby server making the standby server an active server

Parameters

- **url** (*str*) – Base url of CSM server. ex. https://servername:port/CSM/web.
- **tk** (*str*) – Rest token for the CSM server.

Returns JSON String representing the result of the command. 'I' = successful, 'W' = warning, 'E' = error.

3.9 Authorization

This page holds the documentation for the different methods contained in the Authorization folder

3.9.1 Code documentation

pyCSM.authorization.auth.**change_properties**(*property_dictionary*)

Takes a dictionary of properties and the values that user wants to change and changes them in the file.

Parameters

- **property_dictionary** (*dict*) – Dictionary of the keys and values that need
- **file.** (*to be changed in the*) –
- **{"language"** (*ex.*) – "en-UK", "verify":True}

Returns Returns the new properties dictionary.

pyCSM.authorization.auth.**get_properties**()

Returns a dictionary of the current properties and their values set for the file.

pyCSM.authorization.auth.**get_token**(*url*, *username*, *password*)

Retrieves a REST token from the server to be used for future REST commands.

Parameters

- **url** (*str*) – Base url of csm server ex. https://servername:port/CSM/web.
- **username** (*str*) – username for server login.
- **password** (*str*) – password for server login.

Returns Returns a token string to be used to make future rest calls to the given CSM server.

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

INDEX

D

E

G

H

M

P

R

S

T

U

W