
Isogeo - Migrations Toolbelt Documentation

Release 1.0.0

Isogeo

Jan 11, 2022

Contents:

1	Package modules	3
1.1	isogeo_migrations_toolbelt	3
2	Indices and tables	11
	Python Module Index	13
	Index	15

Author Isogeo

Source code <https://github.com/Isogeo/migrations-toolbelt/>

Issues <https://github.com/Isogeo/migrations-toolbelt/issues>

1.1 isogeo_migrations_toolbelt

1.1.1 isogeo_migrations_toolbelt package

Subpackages

isogeo_migrations_toolbelt.backup package

Submodules

isogeo_migrations_toolbelt.backup.backup_manager module

```
class isogeo_migrations_toolbelt.backup.backup_manager.BackupManager (api_client:  
                                                                    iso-  
                                                                    geo_pysdk.isogeo.Isogeo,  
                                                                    out-  
                                                                    put_folder:  
                                                                    str)
```

Bases: object

Backup Manager makes it easy to backup Isogeo data (metadata, contacts, workgroups...). It uses the Isogeo Python SDK to download data asynchronously.

Parameters

- **api_client** (*Isogeo*) – API client authenticated to Isogeo
- **output_folder** (*str*) – path to the folder where to store the exported data

metadata (*search_params: dict, output_format: str = 'json'*) → bool

Backups every metadata corresponding at a search. It builds a list of metadata to export before transmitting it to an async loop.

Parameters

- **search_params** (*dict*) – API client authenticated to Isogeo
- **output_format** (*str*) – format of exported data. Until now, only JSON is available.

Returns True if export reached the end

Return type bool

Example

```
# prepare backup manager
backup_mgr = BackupManager(api_client=isogeo, output_folder="./output")

# build search parameters. For example to filter on two specifics metadata
search_parameters = {
    "query": None,
    "specific_md": [
        METADATA_UUID_1,
        METADATA_UUID_2,
    ],
}

# launch the backup
backup_mgr.metadata(search_params=search_parameters)
```

isogeo_migrations_toolbelt.delete package

Submodules

isogeo_migrations_toolbelt.delete.deleter module

class isogeo_migrations_toolbelt.delete.deleter.**MetadataDeleter** (*api_client:*
iso-
geo_pysdk.isogeo.Isogeo)

Bases: object

Backup Manager makes it easy to backup Isogeo data (metadata, contacts, workgroups...). It uses the Isogeo Python SDK to download data asynchronously.

Parameters

- **api_client** (*Isogeo*) – API client authenticated to Isogeo
- **output_folder** (*str*) – path to the folder where to store the exported data

delete (*metadata_ids_list: list, hard_mode: bool = 0*) → bool

Delete every metadata which UUID appears in metadata_ids_list.

Parameters

- **metadata_ids_list** (*list*) – list of Isogeo Metadata UUID to delete
- **hard_mode** (*bool*) –

Example


```
# prepare deletion manager
md_dltr = MetadataDeleter(api_client=isogeo)

# launch the deletion
md_dltr.delete(metadata_ids_list=li_uuid, hard_mode=1)
```

isogeo_migrations_toolbelt.duplicate package

Submodules

isogeo_migrations_toolbelt.duplicate.duplicator module

class isogeo_migrations_toolbelt.duplicate.duplicator.**MetadataDuplicator** (*api_client: iso-geo_pysdk.isogeo.Isogeo, source_metadata_uuid: uuid.UUID*)

Bases: object

Duplicate metadata. Most used for development purposes.

Parameters

- **api_client** (*Isogeo*) – already authenticated Isogeo client to use to performe API operations
- **source_metadata_uuid** (*UUID*) – UUID of the metadata to be duplicated (source)

duplicate_into_other_group (*destination_workgroup_uuid: str, copymark_catalog: str = None, copymark_title: bool = True, copymark_abstract: bool = True, exclude_catalogs: list = [], exclude_subresources: list = []*) → iso-geo_pysdk.models.metadata.Metadata

Create an exact copy of the metadata source into another workgroup. It can apply some copy marks to distinguish the copy from the original.

Parameters

- **copymark_catalog** (*str*) – add the new metadata to this additionnal catalog. Defaults to None
- **copymark_title** (*bool*) – add a [COPY] mark at the end of the new metadata (default: {True}). Defaults to True
- **copymark_abstract** (*bool*) – add a [Copied from](./source_uuid)] mark at the end of the new metadata abstract. Defaults to True
- **exclude_catalogs** (*list*) – list of catalogs UUID's to not associate to destination metadata

:param list exclude_subresources : list of subressources to be excluded. Must be metadata attributes names

Returns the newly created Metadata

Return type Metadata

Example

```
# instantiate the metadata duplicator
md_source = MetadataDuplicator(
    isogeo=isogeo,
    source_metadata_uuid=environ.get("ISOGEO_METADATA_FIXTURE_UUID")
)
# duplicate it
new_md = md_source.duplicate_into_same_group()
```

duplicate_into_same_group(*copymark_catalog*: str = None, *copymark_title*: bool = True, *copymark_abstract*: bool = True, *exclude_catalogs*: list = [], *switch_service_layers*: bool = False) → isogeo_pysdk.models.metadata.Metadata

Create an exact copy of the metadata source in the same workgroup. It can apply some copy marks to distinguish the copy from the original.

Parameters

- **copymark_catalog** (*str*) – add the new metadata to this additional catalog. Defaults to None
- **copymark_title** (*bool*) – add a [COPY] mark at the end of the new metadata (default: {True}). Defaults to True
- **copymark_abstract** (*bool*) – add a [Copied from](./source_uuid)] mark at the end of the new metadata abstract. Defaults to True
- **exclude_catalogs** (*list*) – list of catalogs UUID's to not associate to destination metadata
- **switch_service_layers** (*bool*) – a service layer can't be associated to many datasets. If this option is enabled, service layers are removed from the metadata source then added to the new one. Defaults to False

Returns the newly created Metadata

Return type Metadata

```
# instantiate the metadata duplicator
md_source = MetadataDuplicator(
    isogeo=isogeo,
    source_metadata_uuid=environ.get("ISOGEO_METADATA_FIXTURE_UUID")
)
# duplicate it
new_md = md_source.duplicate_into_same_group()
```

import_into_other_metadata(*destination_metadata_uuid*: str, *copymark_catalog*: str = None, *copymark_title*: bool = True, *copymark_abstract*: bool = True, *exclude_catalogs*: list = [], *switch_service_layers*: bool = False, *exclude_fields*: list = ['coordinateSystem', 'envelope', 'features', 'geometry', 'name', 'path'], *exclude_subresources*: list = []) → isogeo_pysdk.models.metadata.Metadata

Import a metadata content into another one. It can exclude some fields. It can apply some copy marks to distinguish the copy from the original.

Parameters

- **destination_metadata_uuid** (*str*) – UUID of the metadata to update with source metadata
- **exclude_fields** (*list*) – list of fields to be excluded. Must be attributes names

:param list exclude_subresources : list of subresources to be excluded. Must be metadata attributes names
 :param str copymark_catalog: add the new metadata to this additional catalog. Defaults to None
 :param bool copymark_title: add a [COPY] mark at the end of the new metadata (default: {True}). Defaults to True
 :param bool copymark_abstract: add a [Copied from](./source_uuid) mark at the end of the new metadata abstract. Defaults to True
 :param list exclude_catalogs: list of catalogs UUID's to not associate to destination metadata
 :param bool switch_service_layers: a service layer can't be associated to many datasets. If this option is enabled, service layers are removed from the metadata source then added to the new one. Defaults to False

Returns the updated Metadata

Return type Metadata

```
# TO DO
```

isogeo_migrations_toolbelt.match package

Submodules

isogeo_migrations_toolbelt.match.matcher module

```
class isogeo_migrations_toolbelt.match.matcher.Matcher (api_client: iso-
                                                         geo_pysdk.isogeo.Isogeo,
                                                         source_workgroup_uuid:
                                                         uuid.UUID = "", target_
                                                         get_workgroup_uuid:
                                                         uuid.UUID = "", query: str
                                                         = "")
```

Bases: object

Mathcher use Isogeo Python SDK to establish matches between metadatas according to specified matching criteria. It is useful to generate matching table for metadata migration.

Parameters

- **api_client** (*Isogeo*) – API client authenticated to Isogeo
- **source_workgroup_uuid** (*str*) – the uuid of source metadatas workgroup
- **target_workgroup_uuid** (*str*) – the uuid of target metadatas workgroup

mapping_table (*add_app_url: bool = True, csv_file_path: Union[str, pathlib.Path] = ""*)

Perform the mapping table between previously loaded source and target metadatas and optionally write it into a csv file at the indicated location (csv_file_path)

Parameters

- **add_app_url** (*bool*) – to add fields for the url of the metadatas into app.isogeo.com
- **csv_file_path** (*str*) – the path of the matching table csv file

matching_table (*source_catalog_uuid: uuid.UUID = "", target_catalog_uuid: uuid.UUID = "", target_case: str = 'up', replace_tup: tuple = ("",), missing_string: str = "", additional_string: str = "", additional_string_type: str = 'prefix', add_app_url: bool = True, csv_file_path: Union[str, pathlib.Path] = ""*)

Perform the matching table between previously loaded source and target metadatas according to matching criteria (case, additional_string and additional_string_type) and write it into a csv file at the indicated location (csv_file_path)

Parameters

- **source_catalog_uuid** (*str*) – the uuid of the source metadatas catalog
- **target_catalog_uuid** (*str*) – the uuid of the target metadatas catalog
- **target_case** (*str*) – the case of the target metadatas name (“low” or “up”)
- **replace_tup** (*str*) – (the string to replace, the string to replace with)
- **missing_string** (*str*) – the string to remove from source metadatas name
- **additional_string** (*str*) – the string added to target metadatas name
- **additional_string_type** (*str*) – “prefix” or “suffix”
- **add_app_url** (*bool*) – to add fields for the url of the metadatas into app.isogeo.com
- **csv_file_path** (*str*) – the path of the matching table csv file

write_csv_file (*file_path: pathlib.Path, headers: list, content: list*)

Write the content into specified csv file *file_path* with headers.

isogeo_migrations_toolbelt.readers package

Submodules

isogeo_migrations_toolbelt.readers.reader_csv module

Usage from the repo root folder:

```
python -m migrate_from_excel
```

```
class isogeo_migrations_toolbelt.readers.reader_csv.CsvReader (filename)
```

```
    Bases: csv.DictReader
```

```
    [summary]
```

```
    Arguments: csv {[type]} – [description]
```

```
    close ()
```

isogeo_migrations_toolbelt.readers.reader_excel module

isogeo_migrations_toolbelt.search_replace package

Submodules

isogeo_migrations_toolbelt.search_replace.search_and_replace module

Name: Backup SearchReplaceManager Purpose: Generic module to perform search and replace into metadatas Author: Isogeo

Python: 3.6+

```
class isogeo_migrations_toolbelt.search_replace.search_and_replace.SearchReplaceManager (api
iso-
geo-
ob-
ject
str
=
'me
dat
at-
trib
dich
=
{'al
stra
Nor
'ti-
tle'
Nor
prej
si-
tion
dich
=
Nor
```

Bases: object

Search and replace Manager makes it easy to search into metadata attributes and replace existing value by a new one.

It uses the Isogeo Python SDK to download data asynchronously.

Parameters

- **api_client** (*Isogeo*) – API client authenticated to Isogeo
- **objects_kind** (*str*) – API objects type on which to apply the search replace. Defaults to 'metadata'.
- **attributes_patterns** (*dict*) – dictionary of metadata attributes and tuple of “value to be replaced”, “replacement value”.
- **prepositions** (*dict*) – dictionary used to manage special cases related to prepositions. Structure: {“preposition to be replaced”: “replacement preposition”}

filter_matching_metadata (*isogeo_search_results: list*) → tuple

Filter search results basing on matching patterns.

Parameters **isogeo_search_results** (*MetadataSearch*) – Isogeo search results (*MetadataSearch.results*)

Returns a tuple of objects with the updated attributes

Return type tuple

replacer (*in_text: str, pattern: tuple*) → str

Filter search results basing on matching patterns.

Parameters

- **in_text** (*str*) – text into search a match

- **pattern** (*tuple*) – tuple of str (“to be replaced”, “replacement”)

search_replace (*search_params: dict = {'query': None}, safe: bool = 1*) → dict

It builds a list of metadata to export before transmitting it to an async loop.

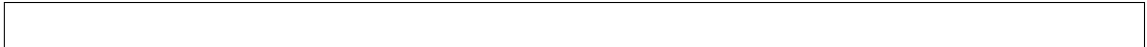
Parameters

- **search_params** (*dict*) – API client authenticated to Isogeo
- **safe** (*bool*) – safe mode enabled or not. In safe mode, the method do not apply modifications online but onyl returns the dictionary with replaced values.

Returns dictionary of metadata with replaced values

Return type dict

Example



isogeo_migrations_toolbelt.search_replace.updater module

Name: Backup Manager Purpose: Generic module to perform backup from Isogeo Author: Isogeo

Python: 3.6+

```
class isogeo_migrations_toolbelt.search_replace.updater.MetadataUpdater (api_client:  
iso-  
geo_pysdk.isogeo.Isogeo,  
meta-  
datas_ready_to_be_updated  
list,  
max_workers=10)
```

Bases: object

batch_updates ()

update (*metadata: isogeo_pysdk.models.metadata.Metadata*)

CHAPTER 2

Indices and tables

- `genindex`
- `modindex`
- `search`

i

- iso19319_xml_fixer, [1](#)
- isogeo_migrations_toolbelt, [3](#)
- isogeo_migrations_toolbelt.backup, [3](#)
- isogeo_migrations_toolbelt.backup.backup_manager,
 [3](#)
- isogeo_migrations_toolbelt.delete, [4](#)
- isogeo_migrations_toolbelt.delete.deleter,
 [4](#)
- isogeo_migrations_toolbelt duplicate, [5](#)
- isogeo_migrations_toolbelt duplicate.duplicator,
 [5](#)
- isogeo_migrations_toolbelt.match, [7](#)
- isogeo_migrations_toolbelt.match.matcher,
 [7](#)
- isogeo_migrations_toolbelt.readers, [8](#)
- isogeo_migrations_toolbelt.readers.reader_csv,
 [8](#)
- isogeo_migrations_toolbelt.search_replace,
 [8](#)
- isogeo_migrations_toolbelt.search_replace.search_and_replace,
 [8](#)
- isogeo_migrations_toolbelt.search_replace.updater,
 [10](#)

B

BackupManager (class in iso-geo_migrations_toolbelt.backup.backup_manager) (module), 3
 batch_updates() (iso-geo_migrations_toolbelt.search_replace.updater.MetadataUpdater method), 10
 isogeo_migrations_toolbelt.backup.backup_manager (module), 3
 isogeo_migrations_toolbelt.delete (module), 4
 isogeo_migrations_toolbelt.delete.deleter (module), 4
 isogeo_migrations_toolbelt.duplicate (module), 5

C

close() (isogeo_migrations_toolbelt.readers.reader_csv.CsvReader method), 8
 CsvReader (class in iso-geo_migrations_toolbelt.readers.reader_csv), 8
 isogeo_migrations_toolbelt.duplicate.duplicator (module), 5
 isogeo_migrations_toolbelt.duplicate.duplicator (module), 5
 isogeo_migrations_toolbelt.match (module), 7
 isogeo_migrations_toolbelt.match.matcher (module), 7
 isogeo_migrations_toolbelt.readers (module), 8

D

delete() (isogeo_migrations_toolbelt.delete.deleter.MetadataDeleter method), 4
 duplicate_into_other_group() (iso-geo_migrations_toolbelt.duplicate.duplicator.MetadataDuplicator method), 5
 duplicate_into_same_group() (iso-geo_migrations_toolbelt.duplicate.duplicator.MetadataDuplicator method), 6
 isogeo_migrations_toolbelt.readers.reader_csv (module), 8
 isogeo_migrations_toolbelt.search_replace (module), 8
 isogeo_migrations_toolbelt.search_replace.search_and_replace (module), 8
 isogeo_migrations_toolbelt.search_replace.updater (module), 10

F

filter_matching_metadatas() (iso-geo_migrations_toolbelt.search_replace.search_and_replace.SearchReplaceManager method), 9
 mapping_table() (iso-geo_migrations_toolbelt.match.matcher.Matcher method), 7

I

import_into_other_metadata() (iso-geo_migrations_toolbelt.duplicate.duplicator.MetadataDuplicator method), 6
 iso19319_xml_fixer (module), 1
 isogeo_migrations_toolbelt (module), 3
 isogeo_migrations_toolbelt.backup (module), 3
 isogeo_migrations_toolbelt.match.matcher.Matcher (class in iso-geo_migrations_toolbelt.match.matcher), 7
 matching_table() (iso-geo_migrations_toolbelt.match.matcher.Matcher method), 7
 metadata() (isogeo_migrations_toolbelt.backup.backup_manager.BackupManager method), 3
 MetadataDeleter (class in iso-geo_migrations_toolbelt.delete.deleter), 4

MetadataDuplicator (class in iso-
geo_migrations_toolbelt.duplicate.duplicator),
5

MetadataUpdater (class in iso-
geo_migrations_toolbelt.search_replace.updater),
10

R

replacer() (isogeo_migrations_toolbelt.search_replace.search_and_replace.SearchReplaceManager
method), 9

S

search_replace() (iso-
geo_migrations_toolbelt.search_replace.search_and_replace.SearchReplaceManager
method), 10

SearchReplaceManager (class in iso-
geo_migrations_toolbelt.search_replace.search_and_replace),
8

U

update() (isogeo_migrations_toolbelt.search_replace.updater.MetadataUpdater
method), 10

W

write_csv_file() (iso-
geo_migrations_toolbelt.match.matcher.Matcher
method), 8