class astroquery.eso.**EsoClass** [source]

Bases: **QueryWithLogin**

Attributes Summary

AUTH_URL	
CALSELECTOR_URL	
DOWNLOAD_URL	
GUNZIP	
QUERY_INSTRUMENT_URL	
ROW_LIMIT	
USERNAME	

Methods Summary

get_associated_files(datasets, *[, mode, ...])	Invoke Calselector service to find calibration files associated to the provided datasets.
get_headers(product_ids, *[, cache])	Get the headers associated to a list of data product IDs
list_instruments(*[, cache])	List all the available instrument-specific queries offered by the ESO archive.
list_surveys(*[, cache])	List all the available surveys (phase 3) in the ESO archive.
login(*[, username, store_password, ...])	Login to the ESO User Portal.
query_apex_quicklooks(*[, project_id, help, ...])	APEX data are distributed with quicklook products identified with a different name than other ESO products.
query_instrument(instrument, *[, ...])	Query instrument-specific raw data contained in the ESO archive.
query_main(*[, column_filters, columns, ...])	Query raw data contained in the ESO archive.
query_surveys(*[, surveys, cache, help, ...])	Query survey Phase 3 data contained in the ESO archive.
retrieve_data(datasets, *[, continuation, ...])	Retrieve a list of datasets form the ESO archive.

Attributes Documentation

AUTH_URL = '*https://www.eso.org/sso/oidc/token*'

CALSELECTOR_URL = '*https://archive.eso.org/calselector/v1/associations*'

DOWNLOAD_URL = '*https://dataportal.eso.org/dataPortal/file/*'

GUNZIP = '*gunzip*'

QUERY_INSTRUMENT_URL = '*http://archive.eso.org/wdb/wdb/eso*'

ROW_LIMIT = *50*

USERNAME = ''

Methods Documentation

get_associated_files(*datasets: List[str]*, ***, *mode: str = 'raw'*, *savexml: bool = False*, *destination: str = None*) → List[str] [source]

Invoke Calselector service to find calibration files associated to the provided datasets.

Parameters: **datasets** : *list of strings*
List of datasets for which calibration files should be retrieved.

mode : *string*
Calselector mode: 'raw' (default) for raw calibrations,
or 'processed' for processed calibrations.

savexml : *bool*
If true, save to disk the XML association tree returned by Calselector.

destination : *string*
Directory where the XML files are saved (default = astropy cache).

Returns: files
List of unique datasets associated to the input datasets.

get_headers(*product_ids*, ***, *cache=True*) [source]

Get the headers associated to a list of data product IDs

This method returns a **Table** where the rows correspond to the provided data product IDs, and the columns are from each of the Fits headers keywords.

Note: The additional column `'DP.ID'` found in the returned table corresponds to the provided data product IDs.

Parameters: **product_ids** : *either a list of strings or a **Column***
List of data product IDs.

cache : *bool*
Defaults to True. If set overrides global caching behavior. See caching documentation.

Returns: **result** : ***Table***
A table where: columns are header keywords, rows are product_ids.

list_instruments(***, *cache=True*) [source]

List all the available instrument-specific queries offered by the ESO archive.

Returns: **instrument_list** : *list of strings*

cache : *bool*
Defaults to True. If set overrides global caching behavior. See caching documentation.

list_surveys(***, *cache=True*)

List all the available surveys (phase 3) in the ESO archive.

latest

Returns: **survey_list** : *list of strings*

cache : *bool*

Defaults to True. If set overrides global caching behavior. See caching documentation.

login(*, *username: str = None*, *store_password: bool = False*, *reenter_password: bool = False*) → bool

Login to the ESO User Portal.

Parameters: **username** : *str, optional*

Username to the ESO Public Portal. If not given, it should be specified in the config file.

store_password : *bool, optional*

Stores the password securely in your keyring. Default is False.

reenter_password : *bool, optional*

Asks for the password even if it is already stored in the keyring. This is the way to overwrite an already stored passwork on the keyring. Default is False.

query_apex_quicklooks(*, *project_id=None*, *help=False*, *open_form=False*, *cache=True*, ***kwargs*) [source]

APEX data are distributed with quicklook products identified with a different name than other ESO products. This query tool searches by project ID or any other supported keywords.

Examples

```
>>> tbl = Eso.query_apex_quicklooks(project_id='093.C-0144')
>>> files = Eso.retrieve_data(tbl['Product ID'])
```

query_instrument(*instrument*, *, *column_filters={}*, *columns=[]*, *open_form=False*, *help=False*, *cache=True*, ***kwargs*) [source]

Query instrument-specific raw data contained in the ESO archive.

Parameters: **instrument** : *string*

Name of the instrument to query, one of the names returned by **list_instruments**.

column_filters : *dict*

Constraints applied to the query.

columns : *list of strings*

Columns returned by the query.

open_form : *bool*

If **True**, opens in your default browser the query form for the requested instrument.

help : *bool*

If **True**, prints all the parameters accepted in `column_filters` and `columns` for the requested `instrument`.

cache : *bool*

Defaults to True. If set overrides global caching behavior. See caching documentation.

latest

Returns: **table** : ***Table***

A table representing the data available in the archive for the specified instrument, matching the constraints specified in `kwargs`. The number of rows returned is capped by the ROW_LIMIT configuration item.

query_main(*, *column_filters={}*, *columns=[]*, *open_form=False*, *help=False*, *cache=True*, ***kwargs*) [source]

Query raw data contained in the ESO archive.

Parameters: **column_filters** : *dict*

Constraints applied to the query.

columns : *list of strings*

Columns returned by the query.

open_form : *bool*

If **True**, opens in your default browser the query form for the requested instrument.

help : *bool*

If **True**, prints all the parameters accepted in `column_filters` and `columns` for the requested `instrument`.

cache : *bool*

Defaults to True. If set overrides global caching behavior. See caching documentation.

Returns: **table** : ***Table***

A table representing the data available in the archive for the specified instrument, matching the constraints specified in `kwargs`. The number of rows returned is capped by the ROW_LIMIT configuration item.

query_surveys(*, *surveys=''*, *cache=True*, *help=False*, *open_form=False*, ***kwargs*) [source]

Query survey Phase 3 data contained in the ESO archive.

Parameters: **survey** : *string or list*

Name of the survey(s) to query. Should beone or more of the names returned by **list_surveys**. If specified as a string, should be a comma-separated list of survey names.

cache : *bool*

Defaults to True. If set overrides global caching behavior. See caching documentation.

Returns: **table** : ***Table*** *or* ***None***

A table representing the data available in the archive for the specified survey, matching the constraints specified in `kwargs`. The number of rows returned is capped by the ROW_LIMIT configuration item. **None** is returned when the query has no results.

retrieve_data(*datasets*, *, *continuation=False*, *destination=None*, *with_calib=None*, *request_all_objects=False*, *unzip=True*, *request_id=None*) [source]

Retrieve a list of datasets form the ESO archive.

Parameters: **datasets** : *list of strings or string*

List of datasets strings to retrieve from the archive.

destination: string

Directory where the files are copied. Files already found in the destination directory are skipped, unless continuation=True. Default to astropy cache.

continuation : *bool*

Force the retrieval of data that are present in the destination directory.

with_calib : *string*

Retrieve associated calibration files: None (default), 'raw' for raw calibrations, or 'processed' for processed calibrations.

unzip : *bool*

Unzip compressed files from the archive after download. `True` by default.

Returns: **files** : *list of strings or string*

List of files that have been locally downloaded from the archive.

Examples

```
>>> dptbl = Eso.query_instrument('apex', pi_coi='ginsburg')
>>> dpids = [row['DP.ID'] for row in dptbl if 'Map' in row['Object']]
>>> files = Eso.retrieve_data(dpids)
```


Page Source Back to Top