Package 'MGPSDK'

January 20, 2025

Type Package

Title Interact with the Maxar 'MGP' Application Programming Interfaces

Version 1.0.0

Description Provides an interface to the Maxar Geospatial Platform (MGP) Application Programming Interface. <https://www.maxar.com/maxar-geospatial-platform>
It facilitates imagery searches using the MGP Streaming Application Programming Interface via the Web Feature Service (WFS) method, and supports image downloads through Web Map Service (WMS) and Web Map Tile Service (WMTS) Open Geospatial Consortium (OGC) methods.
Additionally, it integrates with the Maxar Geospatial Platform Basemaps Application Programming Interface for accessing Maxar basemaps imagery and seamlines.
The package also offers seamless integration with the Maxar Geospatial Platform Discovery Application Programming Interface, allowing users to search, filter, and sort Maxar content, while retrieving detailed metadata in formats like SpatioTemporal Asset Catalog (STAC) and GeoJSON.

License MIT + file LICENSE

Encoding UTF-8

Imports R6, reticulate

RoxygenNote 7.2.3

VignetteBuilder knitr

Suggests knitr, rmarkdown

NeedsCompilation no

Author Nathan Carr [aut, cre, cph]

Maintainer Nathan Carr <nathan.carr@maxar.com>

Repository CRAN

Date/Publication 2023-09-23 15:20:10 UTC

Contents

`Interface` *Interface*

Description

Interface

Interface

Details

Interface class for interacting with WMS, WFS, and WMTS Streaming classes.

This class provides a unified interface to access the WMS, WFS, and WMTS Streaming classes.

The ‘search‘ function performs a search for features within the specified bounding box and/or with a specified filter.

Value

The downloaded file path

Message displaying success and location of downloaded tiles

Public fields

`mgp_sdk` (Optional) An instance of the MGP_SDK Python library. If NULL, a new instance will be created. Default is NULL.

`py_interface` (Optional) An instance of the Interface class from the MGP_SDK Python library. If NULL, a new instance will be created. Default is NULL.

`env_name` = (Optional) The name of the environment where the MGP_SDK Python library is installed. Default is "R-MGP-SDK".

Methods

Public methods:

- `Interface$new()`
- `Interface$streaming_search()`
- `Interface$streaming_download_image()`
- `Interface$streaming_get_full_res_image()`
- `Interface$basemaps_search()`
- `Interface$basemaps_download_image()`
- `Interface$basemaps_download_tiles()`
- `Interface$discovery_stac_search()`
- `Interface$discovery_search_by_audit_fields()`
- `Interface$discovery_get_root_catalog()`
- `Interface$discovery_get_collection_definition()`

- `Interface$discovery_get_all_collections()`
- `Interface$discovery_get_stac_item()`
- `Interface$discovery_get_top_level_sub_catalog()`
- `Interface$discovery_get_sub_catalog()`
- `Interface$discovery_get_sub_catalog_collection_definition()`
- `Interface$clone()`

Method `new()`**:** Initializes the ‘Interface‘ object. Sets up the environment for using the MGP_SDK Python library.

Usage:

```
Interface$new(mgp_sdk = NULL, py_interface = NULL, env_name = "R-MGP-SDK")
```

Arguments:

`mgp_sdk` (Optional) An instance of the MGP_SDK Python library. If NULL, a new instance will be created. Default is NULL.

`py_interface` (Optional) An instance of the Interface class from the MGP_SDK Python library. If NULL, a new instance will be created. Default is NULL.

`env_name` (Optional) The name of the environment where the MGP_SDK Python library is installed. Default is "R-MGP-SDK".

Method `streaming_search()`**:** Perform a search for features within the specified bounding box and/or with a specified filter.

Usage:

```
Interface$streaming_search(
  bbox = NULL,
  filter = NULL,
  shapefile = FALSE,
  csv = FALSE,
  ...
)
```

Arguments:

`bbox` A string indicating the bounding box of the area of interest (miny,minx,maxy,maxx).

`filter` A string containing a CQL filter used to refine the data of the search. Default is NULL.

`shapefile` A logical indicating whether to return a shapefile. Default is FALSE.

`csv` A logical indicating whether to return a CSV file. Default is FALSE.

`...` Additional arguments to pass to the ‘search‘ method.

Returns: If ‘shapefile‘ is TRUE, the function returns a shapefile of all features and associated metadata. If ‘csv‘ is TRUE, the function returns a CSV file. If neither is specified, the function returns a list of features.

Method `streaming_download_image()`**:** Download an image from a WMS or WMTS service This function allows you to download an image from a Web Map Service (WMS) or a Web Map Tile Service (WMTS). You can specify the bounding box, image dimensions, image format, and other parameters to customize the downloaded image.

Usage:

```
Interface$streaming_download_image(
  bbox = NULL,
  srsname = "EPSG:4326",
  height = NULL,
  width = NULL,
  img_format = "jpeg",
  identifier = NULL,
  zoom_level = NULL,
  download = TRUE,
  outputpath = NULL,
  display = FALSE
)
```

Arguments:

`bbox` A vector of four numeric values specifying the bounding box of the image.

`srsname` A string specifying the spatial reference system (SRS) of the bounding box. Default is "EPSG:4326".

`height` The height of the image in pixels.

`width` The width of the image in pixels.

`img_format` A string specifying the image format. Must be one of "jpeg", "png", or "geotiff".

`identifier` A string specifying the identifier of the image.

`zoom_level` An integer specifying the zoom level of the WMTS image.

`download` A logical value indicating whether to download the image (TRUE) or return the raw image data (FALSE).

`outputpath` A string specifying the directory where the downloaded image should be saved.

`display` A logical value indicating whether to display the downloaded image (TRUE) or not (FALSE).

`gridoffsets` A vector of two numeric values specifying the grid offsets of the image.

`...` Additional parameters to be passed to the WMS or WMTS service.

Returns: If ‘download‘ is TRUE, the function returns the filename of the downloaded image. If ‘download‘ is FALSE, the function returns the raw image data as a binary vector.

Method `streaming_get_full_res_image()`**:** This function is a wrapper for a Python function that retrieves full resolution images.

The function downloads an image with the specified feature ID and additional parameters.

Usage:

```
Interface$streaming_get_full_res_image(
  featureid,
  thread_number = 100,
  bbox = NULL,
  mosaic = FALSE,
  srsname = "EPSG:4326",
  outputdirectory = getwd(),
  image_format = "jpeg",
  filename = "Maxar_Download"
)
```

Arguments:

`featureid` A character string representing the unique ID of the feature for which the image is required.

`thread_number` An integer indicating the number of threads to use for the download process. Default is 100.

`bbox` A character string representing the bounding box coordinates in the format 'xmin, ymin, xmax, ymax'. If NULL, the bounding box will be determined based on the feature ID. Default is NULL.

`mosaic` A logical value indicating whether to mosaic the images or not. If TRUE, images covering the defined area will be combined into a single image. Default is FALSE.

`srsname` A character string representing the spatial reference system to be used for the image. Default is 'EPSG:4326'.

`outputdirectory` A character string representing the directory where the image should be saved. If NULL, the image will be saved in the current working directory. Default is NULL.

`image_format` A character string representing the format of the image file to be downloaded. Default is 'jpeg'.

`filename` A character string representing the name of the file to be saved. Default is "Maxar_Download".

Returns: The function returns the result of the Python function call. The nature of this result depends on the Python function implementation.

Method `basemaps_search():` Function searchs using WFS

Usage:

```
Interface$basemaps_search(
  bbox,
  srsname = "EPSG:4326",
  filter,
  shapefile = FALSE,
  csv = FALSE,
  seamlines = FALSE,
  ...
)
```

Arguments:

`bbox` Type:str, Bounding box of the AOI. Comma delimited set of coordinates. (miny,minx,maxy,maxx)

`srsname` Type:str, The desired projection. Defaults to EPSG:4326

`filter` Type: str, CQL filter used to refine the data returned from the search.

`shapefile` Type: bool, Optional Boolean of whether to return in shapefile format. Defaults to false

`csv` Type: bool, Optional Boolean of whether to return in csv format. Defaults to false

`featureprofile` Type: str, Optional. Represents the desired stacking profile. Defaults to account default.

`typename` Type:str, Optional The typename of the desired feature type. Defaults to Finished-Feature.

Method `basemaps_download_image():` Function Downloads a seamline image using the WMS method

Usage:

```
Interface$basemaps_download_image(
  bbox,
  srsname = "EPSG:4326",
  height = NULL,
  width = NULL,
  img_format = "jpeg",
  download = TRUE,
  seamlines = FALSE,
  outputpath
)
```

Arguments:

`bbox` Type:str, Bounding box of the AOI. Comma delimited set of coordinates. (miny,minx,maxy,maxx)

`srsname` Type:str, The desired projection. Defaults to EPSG:4326

`height` Type:int, The vertical number of pixels to return. Defaults to 512

`width` Type:int, The horizontal number of pixels to return. Defaults to 512

`img_format` Type: str, The format of the response image either jpeg, png or geotiff

`download` Type: bool, User option to download file locally. Default True

`outputpath` Type: str Output path must include output format. Downloaded path default is user home path.

`zoom_level` Type: int, The zoom level. Used for WMTS

Method `basemaps_download_tiles()`**:** Function downloads all tiles within a bbox dependent on zoom level

Usage:

```
Interface$basemaps_download_tiles(
  bbox,
  zoom_level,
  srsname = "EPSG:4326",
  img_format = "jpeg",
  seamlines = FALSE,
  outputpath = NULL
)
```

Arguments:

`bbox` Type:str, Bounding box of the AOI. Comma delimited set of coordinates. (miny,minx,maxy,maxx)

`zoom_level` Type: int, The zoom level. Used for WMTS

`srsname` Type:str, The desired projection. Defaults to EPSG:4326

`img_format` Type: str, The format of the response image either jpeg, png or geotiff

`outputpath` Type: str Output path must include output format. Downloaded path default is user home path.

`download` Type: bool, User option to download file locally. Default True

Method `discovery_stac_search()`**:** Returns a list of STAC items

Usage:

```
Interface$discovery_stac_search(...)
```

Arguments:

`collections` (string) = Comma-separated list of collections to search in. Use str format not a Python list

`sub_catalog_id` (string) = Name of the subCatalogId to search in

`sub_catalog_collection` (string) = Used to denote collections inside of sub catalogs

`bbox` (string) = Bounding box in format "minx,miny,maxx,maxy" in WGS84 decimal degrees

`datetime` (string) = Date range filter in ISO 8601 format "start-date/end-date" or exact datetime

`stac_id` (string) = Comma-separated list of STAC item IDs to return. Use str format not a Python list

`intersects` (string) = GeoJSON geometry to search by

`where` (string) = SQL-style WHERE clause for filtering STAC items by properties

`orderby` (string) = SQL-style ORDER BY clause. Only for id and datetime e.g. 'orderby=id'

`limit` (int) = Maximum number of items to return

Method `discovery_search_by_audit_fields()`**:** Retrieve items for a given collectionId by audit fields

Usage:

```
Interface$discovery_search_by_audit_fields(collection_id, ...)
```

Arguments:

`collection_id` (string) = Name of the collection to search e.g. wv01 Required

`audit_insert_date` (string) = Date range filter in ISO 8601 format "start-date/end-date" or exact datetime

`audit_update_date` (string) = Date range filter in ISO 8601 format "start-date/end-date" or exact datetime

`limit` (int) = Maximum number of items to return

Method `discovery_get_root_catalog()`**:** Returns the root STAC Catalog or STAC Collection that is the entry point for users to browse

Usage:

```
Interface$discovery_get_root_catalog(...)
```

Method `discovery_get_collection_definition()`**:** Return a collection definition by collection ID

Usage:

```
Interface$discovery_get_collection_definition(collection_id)
```

Arguments:

`collection_id` (string) = Name of the collection to search e.g. wv01 Required

Method `discovery_get_all_collections()`**:** Return definitions for all collections

Usage:

```
Interface$discovery_get_all_collections(...)
```

Arguments:

`orderby` (string) = SQL-style ORDER BY clause. Only for id and datetime e.g. 'orderby=id ASC' default 'datetime DESC, id ASC' `limit` (int) = Maximum number of items to return

Method `discovery_get_stac_item()`**:** View details about a specific STAC item Dictionary of the desired item's information

Usage:

```
Interface$discovery_get_stac_item(collection_id, item_id)
```

Arguments:

`collection_id` (string) = Name of the collection to search e.g. wv01

`item_id` (string) = Identifier of the desired item

Method `discovery_get_top_level_sub_catalog()`**:** View the available Maxar Sub-Catalogs that can be navigated as a self-contained STAC catalog

Usage:

```
Interface$discovery_get_top_level_sub_catalog(...)
```

Arguments:

`orderby` (string) = SQL-style ORDER BY clause. Only for id and datetime e.g. 'orderby=id ASC' default'datetime DESC, id ASC'

`limit` (int) = Maximum number of items to return

Method `discovery_get_sub_catalog()`**:** View the definition of a Maxar Sub-Catalog

Usage:

```
Interface$discovery_get_sub_catalog(sub_catalog_id)
```

Arguments:

`sub_catalog_id` (string) = Identifier of the sub catalog to view

Method `discovery_get_sub_catalog_collection_definition()`**:** View the definition of a collection that belongs to a Sub-Catalog

Usage:

```
Interface$discovery_get_sub_catalog_collection_definition(
  sub_catalog_id,
  sub_catalog_collection_id
)
```

Arguments:

`sub_catalog_id` (string) = Identifier of the sub catalog to view

`sub_catalog_collection_id` (string) = Identifier of the sub catalog collection to view

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Interface$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Index