KAN Imagery Catalog

Release v0.1.3

Fernando D. Gómez :: Kan Territory & IT

Oct 27, 2023

CONTENTS:

QGIS plugin to get satellite images from different providers.

CHAPTER ONE

API REFERENCE

This page contains auto-generated API reference documentation[1].

1.1 src

This script initializes the plugin, making it known to QGIS.

1.1.1 Subpackages

src.core

Submodules

src.core.catalogs

Catalogs module.

Module Contents

Functions

get_catalog(→ dict)	Get catalog data from a specific provider.
get_catalogs_from_collection(→ list)	Get catalog list from a specific provider.
get_thumbnail(→ dict)	Get catalog thumbnail from a specific provider.
get_quicklook(→ dict)	Get catalog quicklook from a specific provider.
get_download(→ dict)	Get catalog download from a specific provider.
get_download_url(provider)	Get catalog download url from a specific provider.

src.core.catalogs.**get_catalog**(*provider: str*, *host_name: str*, *search_params: dict*) → dict

Get catalog data from a specific provider.

src.core.catalogs.**get_catalogs_from_collection**(*collections: list*, *provider: str*, *search_params: dict*) → list

Get catalog list from a specific provider.

[1] Created with sphinx-autoapi

`src.core.catalogs.`**`get_thumbnail`**(*provider: str*, *host_name: str*, *image_id: str*) → dict

Get catalog thumbnail from a specific provider.

`src.core.catalogs.`**`get_quicklook`**(*provider: str*, *host_name: str*, *image_id: str*) → dict

Get catalog quicklook from a specific provider.

`src.core.catalogs.`**`get_download`**(*provider: str*, *host_name: str*, *search_params: dict*) → dict

Get catalog download from a specific provider.

`src.core.catalogs.`**`get_download_url`**(*provider: str*)

Get catalog download url from a specific provider.

src.core.collections

Collections module.

Module Contents

Functions

get_collections(→ dict)	Get collections from a specific provider.

`src.core.collections.`**`get_collections`**(*provider: str*, *search_params: dict = None*) → dict

Get collections from a specific provider.

src.core.providers

Providers module.

Module Contents

Functions

check_credentials(→ bool)	Check credentials from a specific provider.

`src.core.providers.`**`check_credentials`**(*provider: str*, *credentials: dict*) → bool

Check credentials from a specific provider.

src.core.results

This module contains the functions to create the results layers.

Module Contents

Functions

create_results_layers(provider_name, host, ...)	Create results layers.
create_quicklook(provider_name, host, image_id, layer_name)	Create results layers.

src.core.results.**create_results_layers**(*provider_name*, *host*, *feature_data*, *layer_name*, *group_name*, *footprints_layer*)

Create results layers.

src.core.results.**create_quicklook**(*provider_name*, *host*, *image_id*, *layer_name*)

Create results layers.

src.core.settings

PluginSettings module.

Module Contents

Classes

PluginSettings	PluginSettings class.

class src.core.settings.**PluginSettings**

PluginSettings class.

property provider_settings: dict

property selected_collections: list

property last_cleaning_date: datetime.datetime

language: str = 'english'

back_days: int = 20

cloud_coverage: int = 10

max_catalog_results: int = 10

max_features_results: int = 5

download_path: str = './downloads'

_last_cleaning_date: str | None

_provider_settings: dict

_selected_collections: list

__post_init__()

Init settings.

save()

Save settings.

clean_temporary_files_if_needed()

Checks whether temporary files need to be deleted based on the days before the last activity date and delete files.

update_last_cleaning_date()

Update last activity date.

src.gui

Subpackages

src.gui.helpers

Submodules

src.gui.helpers.forms

Form helpers functions module.

Module Contents

Functions

load_combobox(obj_combobox, key_member, value_member, ...)	Load a QComboBox with provided data.
set_form_stylesheet(parent)	Set stylesheet to a form.
set_elided_text_to_label(label, text)	Set elided text to a QLabel if the text is too long.
set_tooltips(form)	Set tooltips to all QLineEdit widgets in a form.
get_shadow_effect()	Get shadow effect for widgets.
init_tableview(obj_tableview, column_count, row_count, ...)	Initialize tableview with provided data, headers and columns.
load_table_data(obj_table, data[, headers, columns, ...])	Load table with provided data, headers and columns.
check_int_not_empty(txt_widget)	Check if the value is an integer and greater than 0 in a QLineEdit.

src.gui.helpers.forms.**load_combobox**(*obj_combobox*, *key_member*, *value_member*, *lst_data*, *block_signals=False*)

Load a QComboBox with provided data.

src.gui.helpers.forms.**set_form_stylesheet**(*parent*)

Set stylesheet to a form.

src.gui.helpers.forms.**set_elided_text_to_label**(*label*, *text*)

Set elided text to a QLabel if the text is too long.

src.gui.helpers.forms.**set_tooltips**(*form*)

Set tooltips to all QLineEdit widgets in a form.

src.gui.helpers.forms.**get_shadow_effect**()

Get shadow effect for widgets.

src.gui.helpers.forms.**init_tableview**(*obj_tableview*, *column_count*, *row_count*, *headers*, *default_row_height=25*)

Initialize tableview with provided data, headers and columns.

src.gui.helpers.forms.**load_table_data**(*obj_table*, *data*, *headers=None*, *columns=[]*, *fn_custom_items=[]*, *cols_to_stretch=[]*, *cols_width=[]*, *cols_to_hide=[]*, *add_dict_object=False*, *literal_values=False*, *default_row_height=25*, *default_alignment=None*, *checkable_columns=[]*, *num_decimals=2*)

Load table with provided data, headers and columns.

src.gui.helpers.forms.**check_int_not_empty**(*txt_widget*)

Check if the value is an integer and greater than 0 in a QLineEdit.

Submodules

src.gui.custom_widgets

Custom widgets module.

Module Contents

Classes

CustomWidgetListItem	Custom widget for list item.

class src.gui.custom_widgets.**CustomWidgetListItem**(*parent=None*, *provider_name=''*, *host_name=''*, *collection_name=''*, *feature_data=None*, *thumbnail=None*, *acquisition_date=''*, *incidence_angle=0.0*, *cloud_coverage=None*, *image_id=None*, *feature_index=None*, *footprints_layer=None*, *closing_plugin=None*)

Bases: PyQt5.QtWidgets.QWidget, ui.custom_widget_list_item.Ui_CustomWidgetListItem

Custom widget for list item.

property name

property acquisition_date

property incidence_angle

property cloud_coverage

set_thumbnail(*image_bytes*)

Set thumbnail image in label.

download_images()

Download image action.

view_details()

View details action.

get_quicklook()

Get quicklook action.

src.gui.form_base

Module Contents

Classes

FormBase	

class src.gui.form_base.**FormBase**(*parent=None*, *accept_btn=False*, *cancel_btn=False*, *close_btn=True*, *closing_plugin=None*)

Bases: PyQt5.QtWidgets.QDialog

set_control_buttons(*accept_btn*, *cancel_btn*, *close_btn*)

btn_close_click()

src.gui.form_catalog_info

Form Catalog Info Module.

Module Contents

Classes

FormCatalogInfo	Form Catalog Info Class.

class src.gui.form_catalog_info.**FormCatalogInfo**(*parent=None*, *data=None*, *closing_plugin=None*)

Bases: `gui.form_base.FormBase`, `ui.frm_catalog_info.Ui_frm_catalog_info`

Form Catalog Info Class.

normalize_dict_data(*data*)

Normalize data to be loaded in table.

load_data(*data*)

Load data in table.

src.gui.form_default_collections

Collection selection form module.

Module Contents

Classes

FormDefaultCollections	Collection selection form class.

class src.gui.form_default_collections.**FormDefaultCollections**(*parent=None*, *closing_plugin=None*)

Bases: `gui.form_base.FormBase`, `ui.frm_default_collections.Ui_frm_default_collections`

Collection selection form class.

btn_accept_clicked()

Event handler for accept button click.

btn_add_selected_clicked()

Event handler for add selected button click.

btn_remove_selected_clicked()

Event handler for remove selected button click.

btn_filter_results_clicked()

Event handler for filter results button click.

load_filtered_collections(*data*)

Load filtered collections in table.

load_selected_collections()

Load selected collections in table.

src.gui.form_settings

Form settings module.

Module Contents

Classes

FormSettings	Form PluginSettings Class.

class src.gui.form_settings.**FormSettings**(*parent=None*, *closing_plugin=None*)

Bases: gui.form_base.FormBase, ui.frm_settings.Ui_frm_settings

Form PluginSettings Class.

update_cloud_coverage_label()

Update cloud coverage label text.

btn_accept_clicked()

Event handler for accept button click.

txt_default_back_days_text_changed()

Event handler for default back days text changed.

txt_max_catalog_results_text_changed()

Event handler for max catalog results text changed.

txt_max_features_results_text_changed()

Event handler for max features results text changed.

btn_up42_check_credentials_clicked()

Event handler for check credentials button click.

src.services

Submodules

src.services.sentinel_hub

Module for Sentinel Hub API calls

Module Contents

Functions

get_token(client_id, client_secret)	Get token from Sentinel Hub API
get_token_info(client_id)	Get token info from Sentinel Hub API

src.services.sentinel_hub.**get_token**(*client_id*, *client_secret*)

Get token from Sentinel Hub API

src.services.sentinel_hub.**get_token_info**(*client_id*)

Get token info from Sentinel Hub API

src.services.up42

Module for Up42 API calls

Module Contents

Functions

get_token(→ str)	Get token from UP42 API
get_collections()	Get collections from UP42 API
get_catalog(→ dict)	Get catalog data from UP42 API
get_thumbnail(token, host_name, image_id)	Get catalog thumbnail from UP42 API
get_quicklook(token, host_name, image_id)	Get catalog quicklook from UP42 API

Attributes

REQUEST_TIMEOUT
DOWNLOAD_URL

src.services.up42.**REQUEST_TIMEOUT = 120**

src.services.up42.**DOWNLOAD_URL = 'https://console.up42.com/catalog/new-order'**

src.services.up42.**get_token**(*project_id: str*, *api_key: str*) → str

Get token from UP42 API

src.services.up42.**get_collections**()

Get collections from UP42 API

src.services.up42.**get_catalog**(*token: str*, *host_name: str*, *search_params: dict*) → dict

Get catalog data from UP42 API

src.services.up42.**get_thumbnail**(*token: str*, *host_name: str*, *image_id: str*)

Get catalog thumbnail from UP42 API

src.services.up42.**get_quicklook**(*token: str*, *host_name: str*, *image_id: str*)

Get catalog quicklook from UP42 API

`src.services.utils`

Module for API auxiliary functions

Module Contents

Functions

encode_base64(→ str)	Encode a string to Base64

`src.services.utils.`**`encode_base64`**`(`*text_to_encode: str*`)` → str

Encode a string to Base64

`src.ui`

Submodules

`src.ui.custom_widget_list_item`

Module Contents

Classes

Ui_CustomWidgetListItem

Attributes

app

`class` `src.ui.custom_widget_list_item.`**`Ui_CustomWidgetListItem`**

Bases: `object`

`setupUi``(`*CustomWidgetListItem*`)`

`retranslateUi``(`*CustomWidgetListItem*`)`

`src.ui.custom_widget_list_item.`**`app`**

src.ui.frm_catalog_info

Module Contents

Classes

Ui_frm_catalog_info	

Attributes

app	

class src.ui.frm_catalog_info.**Ui_frm_catalog_info**

Bases: object

setupUi(*frm_catalog_info*)

retranslateUi(*frm_catalog_info*)

src.ui.frm_catalog_info.**app**

src.ui.frm_default_collections

Module Contents

Classes

Ui_frm_default_collections	

Attributes

app	

class src.ui.frm_default_collections.**Ui_frm_default_collections**

Bases: object

setupUi(*frm_default_collections*)

retranslateUi(*frm_default_collections*)

src.ui.frm_default_collections.**app**

src.ui.frm_settings

Module Contents

Classes

Ui_frm_settings	

Attributes

app	

class src.ui.frm_settings.**Ui_frm_settings**

Bases: object

setupUi(*frm_settings*)

retranslateUi(*frm_settings*)

src.ui.frm_settings.**app**

src.utils

Submodules

src.utils.constants

Module Contents

Classes

CustomEnum	Generic enumeration.
StyleVariables	Generic enumeration.
MessageType	

Attributes

DEBUG_MODE	
RESULTS_GROUP_NAME	
RESULTS_LAYER_NAME	

src.utils.constants.**DEBUG_MODE = True**

src.utils.constants.**RESULTS_GROUP_NAME = 'kan_imagery_catalog_preview'**

src.utils.constants.**RESULTS_LAYER_NAME = 'kan_imagery_catalog_footprints'**

class src.utils.constants.**CustomEnum**

Bases: enum.Enum

Generic enumeration.

Derive from this class to define new enumerations.

__repr__() → str

Return repr(self).

__str__() → str

Return str(self).

__eq__(*other*)

Return self==value.

classmethod listar()

classmethod to_dict() → dict

classmethod to_key_value_list() → list

class src.utils.constants.**StyleVariables**

Bases: *CustomEnum*

Generic enumeration.

Derive from this class to define new enumerations.

FONT_SIZE_DEFAULT = '12px'

FONT_SIZE_SMALL = '8px'

COLOR_MAIN_DARK = '#6597AF'

COLOR_MAIN_LIGHT = '#F5F5F5'

COLOR_TEXT_DEFAULT = '#2D2D2D'

COLOR_TEXT_LIGHT = '#E8E8E8'

COLOR_BACKGROUND_DEFAULT = '#E8E8E8'

class src.utils.constants.**MessageType**

INFO = 0

WARNING = 1

CRITICAL = 2

SUCCESS = 3

src.utils.debug

Module Contents

Functions

pyqt_set_trace()	

src.utils.debug.**pyqt_set_trace**()

src.utils.encoders

Custom JSON encoders for the application.

Module Contents

Classes

CustomJsonDecimalEncoder	Custom JSON encoder for Decimal objects.

class src.utils.encoders.**CustomJsonDecimalEncoder**(**, skipkeys=False, ensure_ascii=True, check_circular=True, allow_nan=True, sort_keys=False, indent=None, separators=None, default=None*)

Bases: json.JSONEncoder

Custom JSON encoder for Decimal objects.

default(*o*)

Encode Decimal objects as float.

`src.utils.exceptions`

Custom exceptions module.

Module Contents

exception src.utils.exceptions.**PluginError**(*message='', message_type=MessageType.CRITICAL*)

Bases: `Exception`

Base class for custom exceptions.

exception src.utils.exceptions.**ProviderError**(*message='Error to get results from provider.'*)

Bases: *PluginError*

Custom exception for provider related errors.

exception src.utils.exceptions.**HostError**(*message='Error to get results from host.'*)

Bases: *PluginError*

Custom exception for host related errors.

exception src.utils.exceptions.**AuthorizationError**(*message='Error to get results from authorization.'*)

Bases: *PluginError*

Custom exception for authorization related errors.

exception src.utils.exceptions.**SettingsError**(*message='Please check the plugin settings.'*)

Bases: *PluginError*

Custom exception for settings related errors.

exception src.utils.exceptions.**DataNotFoundError**(*message='No data found.'*)

Bases: *PluginError*

Custom exception for data related errors.

`src.utils.general`

General utilities for the KAN Imagery Catalog plugin.

Module Contents

Functions

_get_plugin_name([missing])	Get plugin name from metadata.txt file.
get_plugin_dir()	Get plugin directory.
clean_temporary_files()	

Attributes

PLUGIN_NAME

src.utils.general.**_get_plugin_name**(*missing='KAN Imagery Catalog'*)

Get plugin name from metadata.txt file.

src.utils.general.**get_plugin_dir**()

Get plugin directory.

src.utils.general.**PLUGIN_NAME**

src.utils.general.**clean_temporary_files**()

`src.utils.helpers`

Module Contents

Functions

open_url(→ bool)

src.utils.helpers.**open_url**(*url: str*, *params: dict | None = None*) → bool

`src.utils.qgis_helper`

QGIS helper functions module.

Module Contents

Functions

`get_bounding_box_canvas`()	Get bounding box from canvas.
`get_bounding_box_selected_feature`(layer_name)	Get bounding box from selected feature.
`get_single_polygon_layers`()	Get single polygon layers from current project.
`save_setting`(setting_name, value)	Save setting.
`read_setting`(setting_name[, default_value])	Read setting.
`save_json_setting`(setting_name, value)	Save json setting.
`read_json_setting`(setting_name[, default_value])	Read json setting.
`error_message`(title, text)	Show error message.
`warning_message`(title, text)	Show warning message.
`info_message`(title, text)	Show info message.
`success_message`(title, text)	Show success message.
`get_or_create_group`(group_name)	Get or create group.
`get_layer_by_name`(layer_name)	Get layer by name.
`get_or_create_footprints_layer`(layer_name, group_name)	Get or create footprints layer.
`get_current_crs`()	Get crs from active layer.
`create_layer`(layer_name, group_name)	Create a layer.
`add_feature_to_layer`(feature_data, feature_id, layer)	Add feature to layer.
`create_quicklook_layer`(layer_name, group_name, ...[, ...])	Create a layer with a quicklook image.

src.utils.qgis_helper.**get_bounding_box_canvas**()

Get bounding box from canvas.

src.utils.qgis_helper.**get_bounding_box_selected_feature**(*layer_name*)

Get bounding box from selected feature.

src.utils.qgis_helper.**get_single_polygon_layers**()

Get single polygon layers from current project.

src.utils.qgis_helper.**save_setting**(*setting_name*, *value*)

Save setting.

src.utils.qgis_helper.**read_setting**(*setting_name*, *default_value=None*)

Read setting.

src.utils.qgis_helper.**save_json_setting**(*setting_name*, *value*)

Save json setting.

src.utils.qgis_helper.**read_json_setting**(*setting_name*, *default_value=None*)

Read json setting.

src.utils.qgis_helper.**error_message**(*title*, *text*)

Show error message.

src.utils.qgis_helper.**warning_message**(*title*, *text*)

Show warning message.

src.utils.qgis_helper.**info_message**(*title*, *text*)

Show info message.

`src.utils.qgis_helper.`**`success_message`**`(`*title*, *text*`)`

Show success message.

`src.utils.qgis_helper.`**`get_or_create_group`**`(`*group_name*`)`

Get or create group.

`src.utils.qgis_helper.`**`get_layer_by_name`**`(`*layer_name*`)`

Get layer by name.

`src.utils.qgis_helper.`**`get_or_create_footprints_layer`**`(`*layer_name*, *group_name*`)`

Get or create footprints layer.

`src.utils.qgis_helper.`**`get_current_crs`**`()`

Get crs from active layer.

`src.utils.qgis_helper.`**`create_layer`**`(`*layer_name*, *group_name*`)`

Create a layer.

`src.utils.qgis_helper.`**`add_feature_to_layer`**`(`*feature_data*, *feature_id*, *layer*`)`

Add feature to layer.

`src.utils.qgis_helper.`**`create_quicklook_layer`**`(`*layer_name*, *group_name*, *feature*, *crs*, *layer_type='Polygon'*, *image_path=None*`)`

Create a layer with a quicklook image.

1.1.2 Submodules

`src.kan_imagery_catalog`

Module Contents

Classes

`KANImageryCatalog`	QGIS Plugin Implementation.

`class` `src.kan_imagery_catalog.`**`KANImageryCatalog`**`(`*iface*`)`

QGIS Plugin Implementation.

`tr``(`*message*`)`

Get the translation for a string using Qt translation API.

We implement this ourselves since we do not inherit QObject.

Parameters

`message` (*`str, QString`*) – String for translation.

Returns

Translated version of message.

Return type

QString

add_action(*icon_path*, *text*, *callback*, *enabled_flag=True*, *add_to_menu=True*, *add_to_toolbar=True*, *status_tip=None*, *whats_this=None*, *parent=None*)

Add a toolbar icon to the toolbar.

Parameters

- **icon_path** (*str*) – Path to the icon for this action. Can be a resource path (e.g. ':/plugins/foo/bar.png') or a normal file system path.
- **text** (*str*) – Text that should be shown in menu items for this action.
- **callback** (*function*) – Function to be called when the action is triggered.
- **enabled_flag** (*bool*) – A flag indicating if the action should be enabled by default. Defaults to True.
- **add_to_menu** (*bool*) – Flag indicating whether the action should also be added to the menu. Defaults to True.
- **add_to_toolbar** (*bool*) – Flag indicating whether the action should also be added to the toolbar. Defaults to True.
- **status_tip** (*str*) – Optional text to show in a popup when mouse pointer hovers over the action.
- **parent** (*QWidget*) – Parent widget for the new action. Defaults None.
- **whats_this** – Optional text to show in the status bar when the mouse pointer hovers over the action.

Returns

The action that was created. Note that the action is also added to self.actions list.

Return type

QAction

initGui()

Create the menu entries and toolbar icons inside the QGIS GUI.

onClosePlugin()

Cleanup necessary items here when plugin dockwidget is closed

unload()

Removes the plugin menu item and icon from QGIS GUI.

run()

Run method that loads and starts the plugin

src.kan_imagery_catalog_dock

Module Contents

Classes

KANImageryCatalogDock	KANImageryCatalogDock Dockwidget class.

class `src.kan_imagery_catalog_dock.`**`KANImageryCatalogDock`**(*parent=None*)

Bases: `PyQt5.QtWidgets.QDockWidget`, `FORM_CLASS`

KANImageryCatalogDock Dockwidget class.

`closing_plugin`

`closeEvent`(*event*)

Run close plugin event.

`txt_max_catalog_results_text_changed`()

Event handler for textbox ‘txt_max_catalog_results’.

`update_cloud_coverage_label`()

Event handler for slider ‘slider_cloud_coverage’.

`chk_search_by_dataframe_update`()

Event handler for checkbox ‘chk_search_by_dataframe’.

`btn_update_layers_list_clicked`()

Event handler for button ‘btn_update_layers_list’.

`btn_settings_clicked`()

Event handler for button ‘btn_settings’.

`btn_select_catalogs_clicked`()

Event handler for button ‘btn_select_catalogs’.

`btn_sort_results_clicked`()

Event handler for button ‘btn_sort_results’.

`btn_show_hide_search_area_clicked`()

Event handler for button ‘btn_show_hide_search_area’.

`btn_show_hide_catalogs_clicked`()

Event handler for button ‘btn_show_hide_catalogs’.

`btn_show_hide_filters_clicked`()

Event handler for button ‘btn_show_hide_filters’.

`btn_get_data_clicked`()

Event handler for button ‘btn_get_data’.

`get_results`(*layer_name*, *cloud_coverage*, *date_from*, *date_to*, *max_catalog_results*)

Get results from selected catalogs with selected filters.

`add_item_to_results`(*provider_name*, *host_name*, *collection_name*, *feature_data*, *acquisition_date*, *incidence_angle*, *cloud_coverage*, *image_id*, *feature_index*)

Add item to results list.

`add_feature_to_footprints_layer`(*feature_data*, *footprint_id*)

src.plugin_upload

This script uploads a plugin package to the plugin repository. Authors: A. Pasotti, V. Picavet git sha : $TemplateVCS-Format

Module Contents

Functions

main(parameters, arguments)	Main entry point.
hide_password(url[, start])	Returns the http url with password part replaced with '*'.

Attributes

PROTOCOL
SERVER
PORT
ENDPOINT
VERBOSE
parser

src.plugin_upload.**PROTOCOL = 'https'**

src.plugin_upload.**SERVER = 'plugins.qgis.org'**

src.plugin_upload.**PORT = '443'**

src.plugin_upload.**ENDPOINT = '/plugins/RPC2/'**

src.plugin_upload.**VERBOSE = False**

src.plugin_upload.**main**(*parameters*, *arguments*)

Main entry point.

Parameters

- **parameters** – Command line parameters.
- **arguments** – Command line arguments.

src.plugin_upload.**hide_password**(*url*, *start=6*)

Returns the http url with password part replaced with '*'.

Parameters

- **url** (*str*) – URL to upload the plugin to.

- **start** (*int*) – Position of start of password.

`src.plugin_upload.parser`

src.resources

Module Contents

Functions

qInitResources()
qCleanupResources()

Attributes

qt_resource_data
qt_resource_name
qt_resource_struct_v1
qt_resource_struct_v2
qt_version
rcc_version

```
src.resources.qt_resource_data =
b'\x00\x00\x16\x8b\x89PNG\r\n\x1a\n\x00\x00\x00\rIHDR\x00\x00\x00 \x00\x00\x00...
```

```
src.resources.qt_resource_name = b'\x00\x08\naZ\xa7\x00i\x00c\x00o\x00n\x00.\x00p\x00n\
x00g\x00\t\nlxC\x00r\x00e\x00s\x00o\x00u\x0...
```

```
src.resources.qt_resource_struct_v1 = b'\x00\x00\x00\x00\x00\x02\x00\x00\x00\x02\x00\x00\
x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0...
```

```
src.resources.qt_resource_struct_v2 = b'\x00\x00\x00\x00\x00\x02\x00\x00\x00\x02\x00\x00\
x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0...
```

`src.resources.qt_version`

`src.resources.rcc_version = 1`

`src.resources.qInitResources()`

`src.resources.qCleanupResources()`

1.1.3 Package Contents

Functions

classFactory(iface)	Load KANImageryCatalog class from file KANImageryCatalog.

`src.`**`classFactory`**`(`*iface*`)`

Load KANImageryCatalog class from file KANImageryCatalog.

Parameters

`iface` (*`QgsInterface`*) – A QGIS interface instance.

PYTHON MODULE INDEX

INDEX