GEBOT Documentation

Release 1.0.0

Chirag Padubidri

Dec 20, 2023

CONTENTS:

CHAPTER

ONE

INDICES AND TABLES

- genindex
- modindex
- search

IMAGE DOWNLOADER

ImageDownloader is a class designed for downloading Google Earth images based on coordinates (Longitude, Latitude) retrieved from a CSV database. The images are centered around the specified coordinates.

To set up the bot, ensure that the 'getLoc.py' script has been executed, and the 'config.json' file is located in the './resources' folder. The 'config_path' parameter in the constructor allows customization of the configuration file path.

In the current version, the bot is equipped with download logic, a notification handler (to send the status to the registered email on failure), and a status window (to display the bot's status).

param config_path
The file path to the configuration file (default is './resources/config.json').

type config_path
str, optional

Examples

Initialize the ImageDownloader with a custom configuration file path:

```
>>> downloader = ImageDownloader(config_path='./custom_config/config.json')
```

gebot.ImageDownloader.**__init__**()

Initialize ImageDownloader class.

The configuration file contains all the required parameters to run this class.

Parameters
config_path (*str, optional*) – Path to the configuration file (default: './resources/config.json').

gebot.ImageDownloader.**download_image**(*coord*, *filename*, *hover_time=4*, *step_sleep=2*)

Download an image from input coordinates and save it with the given filename. This method gives more control to the user; the user can download a single image and save it with a user-defined filename.

Parameters

- **coord** (*str*) – Centre coordinates of the image.
- **filename** (*str*) – Name of the file to be saved.

- **hover_time** (*int, optional*) – Time to sleep when GE pro is hovering (default: 4).
- **step_sleep** (*int, optional*) – Time to sleep (default: 2).

gebot.ImageDownloader.**download_images**(*latitude*, *longitude*, *img_id*, *sleep_time=0*, *sleep_after=25*)

Download multiple images from coordinates. This method takes a list of parameters and downloads them; the user has less control over the filename but it is faster.

Parameters

- **latitude** (*list*) – List of latitudes.
- **longitude** (*list*) – List of longitudes.
- **img_id** (*list*) – List of image IDs.
- **sleep_time** (*int, optional*) – Time to sleep (default: 0).
- **sleep_after** (*int, optional*) – Sleep after a certain number of downloads to avoid banning. (default: 25).

gebot.ImageDownloader.**__check_download_complete__**(*filename*)

Check if the download is complete for a specific file on the given save path. This is an internal method to check the status of the download. Once the download is completed, it sends a trigger flag to the bot to continue downloading. Additionally, if the bot gets into a 'stalled' state and stops downloading, it sends an email to the registered user.

Parameters

filename (*str*) – Name of the file to check.

gebot.ImageDownloader.**__update_status__**()

Internal method. This method is used to update the status on the notification window of the bot after each image download.

gebot.ImageDownloader.**__get_status__**()

Internal method for updating the status. This method calculates various variables to be displayed on the status window.

Returns

A dictionary containing status-related information.

- 'status' (str): The current status of the operation.
- 'speed' (str): The processing speed, represented as seconds per image.
- 'expected_finish_time' (str): The estimated time for completion in days, hours, and minutes.
- 'remaining_images' (str): The number of remaining images to process.
- 'time_elapsed' (str): The time elapsed in days, hours, and minutes since the start of the operation.

Return type

dict

Note: The 'speed' is calculated as the time taken per image, and 'expected_finish_time' and 'time_elapsed' are formatted in days, hours, and minutes.

---

gebot.ImageDownloader.**__sec2dhm__**(*duration_seconds*)

This is an internal helper method to calculate days, hours, and minutes from the input seconds.

Parameters
duration_seconds (*int*) – Time in seconds.

Returns
A list with days, hours, and minutes for input seconds. - (days, hours, minutes) (list): Time in Days, Hours, and Minutes

Return type
list

Location Getter

This class is designed to facilitate the manual retrieval and storage of Google Earth Pro GUI button locations, and update the configuration data in 'config.json'. The script serves as a setup script that must be run before configuring GEBOT. It guides the user through the necessary steps to obtain the required configuration file.

param None

getLoc.LocationGetter.**location_report**

A dictionary to store mouse locations for different elements.

Type
dict

getLoc.LocationGetter.**__init__**()

Initializes the LocationGetter.

Parameters
None –

Return type
None

getLoc.LocationGetter.**get_location**(*message*)

Retrieves the mouse location after user input.

Parameters
message (*str*) – The message to display to the user.

Returns
x and y coordinates of the mouse position.

Return type
tuple

getLoc.LocationGetter.**collect_locations**()

Collects mouse locations for the search bar, uncheck, save image, and save button.

Parameters
None –

getLoc.LocationGetter.**print_location_report**()

Saves the location report as a JSON file and prints the location report.

Parameters
None –

SendEmail

A class to send email notifications when the GE bot process is stopped.

notificationHandler.SendEmail.__init__()

Initializes the SendEmail object.

Parameters

- **config_path** (*str, optional*) – The path to the configuration file (default is './resources/config.json').
- **credData** (*str, optional*) – The path to the credential data file (default is "./resources/cred.dat").

Return type
None

notificationHandler.SendEmail.process_stopped()

Sends email notifications when the GE bot process is stopped.

For each email ID in the configured list, connects to the SMTP server, logs in, and sends a notification email.

Parameters
None –

Return type
None

GEBotInfoDisplay

A class for creating and updating a GUI display for GE Bot information using tkinter.

statusIndicator.GEBotInfoDisplay.__init__()

Initializes the GEBotInfoDisplay object with the specified Tkinter root window.

Parameters
(tk.Tk) (*root*) – The Tkinter root window to which the display will be attached.

statusIndicator.GEBotInfoDisplay.create_label()

Creates a labeled display area for a specific information category.

Parameters

- **text** (*str*) – The label text.
- **color** (*str*) – The color of the label text.
- **variable** (*tk.StringVar*) – The Tkinter StringVar associated with the displayed information.

statusIndicator.GEBotInfoDisplay.update_info()

Updates the displayed information based on the provided status dictionary. :param status_dic: A dictionary containing information about GE Bot status. :type status_dic: dict

Geotagger

A class for geotagging images and saving them in TIFF format with spatial information. This should be used only to the images downloaded via GEBOT.

Note: Geotagger class should be run using main function. It can take PNG/JPG files and georeference them to TIFF format.

---

param inputpath
Path to the image folder.

type inputpath
str/path

param outputpath
Path of the folder where the georeferenced images should be saved. If set to 'None', a new folder will be created in the imagePath with '_GEOTAGGED' appended to the path.

type outputpath
str/path

param start
The starting number for the filenames of the images in the folder for georeferencing. If set to 'None', the function will start from the beginning (start=0).

type start
int

param stop
The ending number for the filenames of the images in the folder for georeferencing. If set to 'None', the function will perform the action for all the items in the folder.

type stop
int

The 'start' and 'stop' parameters are helpful if we want to terminate the process in the middle and restart it later.

Examples

```
>>> python georef.py --inputpath path/to/image_folder --outputpath path/to/save_folder --
↪start 10 --stop 20
```

georef.Geotagger.__init__()

Initializes the Geotagger object.

Parameters

- **filepath** (*str*) – The path to the image file.
- **center_coord** (*tuple*) – Tuple containing latitude and longitude values of the image center.
- **savepath** (*str*) – The path to the folder where geotagged images will be saved.
- **pixXRES** (*float*) – Pixel resolution in the X direction.
- **pixYRES** (*float*) – Pixel resolution in the Y direction.

Return type
None

georef.Geotagger.lat_long()

This is a local method to calculates new latitude and longitude with the given offsets.

Parameters

- **lat** (*float*) – Latitude.
- **lon** (*float*) – Longitude.
- **dn** (*float*) – Offset in the north direction.
- **de** (*float*) – Offset in the east direction.

Returns
Tuple containing new latitude, longitude, and altitude (0).

Return type
tuple

georef.Geotagger.**output_corners**()

Computes the corners of the geotagged image based on center coordinates and pixel resolutions.

Parameters

- **lat** (*float*) – Latitude of the image center.
- **lon** (*float*) – Longitude of the image center.
- **width** (*int*) – Width of the image in pixels.
- **height** (*int*) – Height of the image in pixels.
- **offset1** (*float*) – Offset in the north direction.
- **offset2** (*float*) – Offset in the east direction.

Returns
Array containing the coordinates of the image corners.

Return type
numpy.ndarray

georef.Geotagger.**name2latlong**()

Extracts latitude and longitude from the image filename.

Parameters
filename (*str*) – The name of the image file.

Returns
Tuple containing latitude and longitude.

Return type
tuple

georef.Geotagger.**getcoord**()

Computes the geotagged image's bounding box and corner coordinates.

Parameters
None –

Returns
Tuple containing the bounding box coordinates (north, south, west, east) and the image.

Return type
tuple

georef.Geotagger.**geotag**()

Geotags the image and saves it in TIFF format.

PYTHON MODULE INDEX

g

n

s

INDEX

Symbols

C

D

G

L

M

N

O

P

S

U